Formal Analysis of an eHealth Protocol

Naipeng Dong*, Hugo Jonker, and Jun Pang

Faculty of Sciences, Technology and Communication,
University of Luxembourg, Luxembourg

Abstract. Given the sensitive nature of health data, security and privacy of eHealth systems is of prime importance. Properties like secrecy, authentication, anonymity, and untraceability need to be satisfied. However, only satisfying these properties is not sufficient in case users can reveal private information to the adversary. For instance, a pharmaceutical company may bribe or coerce a pharmacist to reveal information which breaks a doctor's privacy. Therefore, new privacy properties are required: *enforced prescribing-privacy*, *independency of prescribing-privacy*, and *independency of enforced prescribing-privacy*. In this paper, we identify and formalise these new properties. Moreover we take an eHealth protocol (DLVV08), which is proposed for practical use, as a case study, and study to what extent all these properties are satisfied by the DLVV08 protocol. Finally, we address found ambiguities and flaws and propose suggestions for fixing them.

1 Introduction

Generally speaking, the term of eHealth means applying communicating electronic components (such as: the Internet, smart cards, digital medical equipment, etc.) to support and improve health care [1]. Nowadays, eHealth systems are more and more involved in everyday life. The use of electronic components raises security and privacy issues due to the sensitive nature of health data. To ensure security and privacy in ehealth, much research has been done. In the literature, security and privacy are often seen as an access control problems [2–4]. Seldom attention has been drawn to security and privacy of communication between components in eHealth systems. Even access control policy is perfectly applied, security and privacy may be violated during communication. For example, a message containing the medical history of a patient may be intercepted in transit. Therefore, in this paper, we consider security and privacy of the involved parties with respect to an outsider, the Dolev-Yao adversary [5], who controls the communication network (i.e. the adversary can observe, block, create and alter information).

Security and privacy of communication are mainly achieved by employing cryptographic communication protocols. However, it is well known that designing cryptographic protocols is error-prone. The claims of an eHealth protocol must be verified before the protocol is used in practice. Without verifying that a

* Supported by a grant from the Fonds National de la Recherche (Luxembourg).

protocol satisfies its security claims, subtle flaws may go undiscovered. Time and again, formal analysis has uncovered flaws in protocols that claimed to be secure (e.g., voting systems [6, 7] have been broken [8, 9]). To be able to verify whether a protocol satisfies security and privacy requirements, a necessary step is to give explicit formal definitions of security and privacy properties. Many security and privacy properties have been defined: secrecy, authentication, anonymity and untraceability. We refer to these properties as regular security and privacy properties. However, it is not sufficient to only satisfy these properties. In case a user may be bribed or coerced to reveal her information, she may harm privacy of herself or others. For example, a pharmaceutical company may bribe doctors to prescribe only their medicines. Therefore, we consider not only privacy with respect to a Dolev-Yao adversary, but also privacy in the presence of an active coercer – someone who is bribing or threatening parties to reveal their privacy. We refer to these properties as enforced privacy properties. This includes enforced prescribing-privacy (preventing doctor bribes), independency of prescribing-privacy (preventing other party bribes to break a doctor's privacy), and independency of enforced prescribing-privacy (preventing both doctor bribes and other party bribes to break a doctor's privacy).

We take the DLVV08 eHealth protocol as a case study [10], for two reasons: first, the protocol claims to satisfy several security and privacy properties of patients and doctors, including enforced prescribing-privacy and prescribing-privacy independent of pharmacist; second, the protocol was proposed for actual use for the Belgian health care system, therefore, the analysis has practical merit. We analyse the following properties of the protocol: standard secrecy of patients' secret information, standard secrecy of doctors' secret information, authentication of a patient, authentication of a doctor, patient and doctor anonymity, patient and doctor untraceability, enforced prescribing-privacy, prescribing-privacy independent of pharmacist and enforced prescribing-privacy independent of pharmacist. We address ambiguities and flaws found during the verifications, propose ways to fix them and update the protocol to satisfy all these properties.

We use the applied pi calculus [11] to formally define these security and privacy properties and model the DLVV08 protocol. The applied pi calculus is suitable for modelling concurrent systems, flexible to define cryptographic primitive and supported by the model checker ProVerif [12]. Once the properties are formally defined, we can verify them automatically on models of protocols using ProVerif.

Contributions. We identify enforced privacy requirements in eHealth systems and formally defined them in the applied pi calculus. Then, we model a practical protocol in the applied pi which involves non-trivial cryptographic primitives. Next, we analyse enforced privacy properties of the protocol, as well as regular security and privacy properties, address ambiguities and flaws, and propose suggestions for fixing them.

Organisation. In the next section, we briefly introduce notations used in the applied pi calculus. After that, privacy properties are formally defined in the applied pi calculus in Section 3. This includes prescribing-privacy, enforced prescribing-privacy, pharmacist-enforced privacy and doctor enforced privacy. Next, we briefly describe the DLVV08 protocol in Section 5, and describe, in details, the modelling of the protocol in Section 6. Then we present the analysis of security and privacy properties of the protocol in Section 7, as well as fixing ambiguities and flaws. Next, we update the protocol to satisfy the analysed properties in Section 8. Finally, we present conclusions and future works in Section 9.

2 The applied pi calculus

The applied pi calculus is a language designed for modelling and analysing security protocols [11]. It assumes an infinite set of names, an infinite set of variables and a set of functions. Names are used to model channels and data, variables are used to model received data, and functions are used to model cryptographic primitives in our protocol model. The applied pi calculus defines a *term* as a name, or a variable, or a function applied on other terms. Terms are used to model messages. The equational theory E defines the equations on terms.

A protocol is modelled as different principals running in parallel. The behaviour of a principal is modelled as processes. A process is defined as in Figure 1. A process is defined as either be an empty process, or sub-processes running in parallel, or replication of a sub-process, or restrict a name to a process, or a conditional statement evaluation, or an input action, or an output action. Extended processes add variable restrictions and active substitution.

P, Q, R ::=		plain processes	
	0		null process
	$P \mid Q$		parallel composition
	$!P$		replication
	$\nu n.P$		name restriction
	if $M =_E N$ then P else Q		conditional
	in$(u, x).P$		message input
	out$(u, M).P$		message output
A, B, C ::=		extented processes	
	P		plain process
	$A \mid B$		parallel composition
	$\nu n.A$		name restriction
	$\nu x.A$		variable restriction
	$\{M/x\}$		active substitution

Fig. 1. Applied pi calculus grammar

Semantics of the applied pi calculus are broken down into three parts: first, structural equivalence, which defines equivalence relations between two processes which only differ in structure; second, internal reduction ($\rightarrow$), which defines subprocess communication rules, and *if-then-else* evaluation rules; third, labelled reduction ($\xrightarrow{\alpha}$), which defines reduction rules to model the communication between the adversary (the context) and the protocol. For more details, see [11].

Notation and terminology. In this paper, we use the following notation and terminology: we use "$P\{M/x\}$" to denote syntactical replacing x with M in process P, and it is the same as "let x=M in P". We use $=_E$ to denote term equivalence relations introduced by equational theory E. A name or a variable is free if it is not delimited by restriction and by inputs. The set of free names, the set of free variable, the set of bound names and the set of bound variables of a process A are denoted as $\mathsf{fn}(A)$, $\mathsf{fv}(A)$, $\mathsf{bn}(A)$ and $\mathsf{bv}(A)$, respectively. A process is closed if it does not contain free variables. A context is defined as a process with a hole, and we can put any process in the hole. An evaluation context is a context whose hole is not in the scope of a replication, a conditional, an input, or an output. A term is ground when it does not contain variables. The frame of a process is the static knowledge revealed to the adversary, which is defined as an extended process composed by parallel compositions of active substitutions and restrictions. The domain of a frame is the set of variables in active substitutions (domain of frame ψ is denoted as $\mathsf{dom}(\psi)$). The start symbol in $\rightarrow *$ denotes that the number of $\rightarrow$ is zero or more .

The applied pi calculus defines relations on processes which can be used to model some security and privacy properties. One of the relations is observational equivalence. The intuition is that two processes are not distinguishable for the adversary. In practice, observational equivalence is hard to use, because of the quantification over contexts. Therefore, labelled bisimilarity is introduced. Labelled bisimilarity is easier to reason manually and automatically. Note that labelled bisimilarity and observational equivalence coincide [11].

Two notations are used in labelled bisimilarity: *static equivalence* ($\approx_s$) and *labelled bisimilarity* ($\approx_\ell$). Static equivalence compares the static states of processes (represented by their frames), while labelled bisimilarity examines their dynamic behaviour.

Definition 1 (Static equivalence [11]). *Two closed frames ψ and ϕ are statically equivalent, $\psi \approx_s \phi$, if*

1. $\mathsf{dom}(\psi)$=$\mathsf{dom}(\phi)$
2. $\forall$ *terms M and N, $(M =_E N)$ in $\psi \Leftrightarrow (M =_E N)$ in ϕ.*

Definition 2 (Labelled bisimilarity [11]). Labelled bisimilarity ($\approx_\ell$) *is defined as the largest symmetric relation $\mathcal{R}$ on closed extended processes, such that process $A \mathcal{R} B$ implies:*

1. $A \approx_s B$;
2. *if $A \rightarrow A'$ then $B \rightarrow^* B'$ and $A' \mathcal{R} B'$ for some B';*
3. *if $A \xrightarrow{\alpha} A'$ and $\mathsf{fv}(\alpha) \subseteq \mathsf{dom}(A)$ and $\mathsf{bn}(\alpha) \cap \mathsf{fn}(B) = \emptyset$; then $B \rightarrow^* \xrightarrow{\alpha} \rightarrow^* B'$ and $A' \mathcal{R} B'$ for some B'.*

The adversary. As in the applied pi calculus, the adversary (Dolev-Yao adversary [5]) controls the whole network: listening, blocking, creating, injecting messages, and applying cryptographic primitives. Notice that normally dishonest users are considered as part of the adversary. However, users who are coerced or bribed are not part of the adversary, for the reason that the adversary does not fully trust the coerced or bribed users unless they can prove his information.

3 Enforced privacy properties

A protocol *EHP* is modelled as roles running in parallel in the applied pi calculus. An eHealth protocol *EHP* is a n-role protocol of the form:

$$EHP = \nu\tilde{m}.init.(!R_1 \mid \ldots \mid !R_n).$$

Particularly, there is a doctor role R_{dr} of the form:

$$R_{dr} = \nu\mathtt{Id}_{dr}.init_{dr}.!P_{dr}, where\ P_{dr} = \nu\mathtt{presc}.main_{dr},$$

a patient role R_{pt} of the form:

$$R_{dr} = \nu\mathtt{Id}_{pt}.init_{pt}.!P_{pt},$$

and a pharmacist role R_{ph}. For instance, the DLVV08 protocol, which contains mainly five roles, is modelled as follows.

$$DLV = \nu\tilde{m}.init.(!R_{pt} \mid !R_{dr} \mid !R_{ph} \mid !R_{mpa} \mid !R_{hii})$$

Processes R_{pt}, R_{dr}, R_{ph}, R_{mpa}, R_{hii} model behaviour of patients, doctors, pharmacists, medicine prescription administrators, and health insurance institutions, respectively. The replication ! in front of each process represents unbounded number of instantiations of each role. Process $\nu\tilde{m}.init$ generates and distributes initial knowledge of the whole process. Processes *init* $init_{pt}$ and $init_{pt}$ are sequential. We define context $\mathcal{C}$ as honest instances of roles running in parallel. For instance, a context in the DLVV08 protocol is:

$$\mathcal{C}_{dlv} = \nu\tilde{m}.init.(!R_{pt} \mid !R_{dr} \mid !R_{ph} \mid !R_{mpa} \mid !R_{hii} \mid \_).$$

3.1 Prescribing-privacy

A doctor's prescription behaviour needs to be protected. Straightforwardly, if a doctor's prescription is private, the adversary cannot tell the doctor's prescription behaviour. In some cases, for example DLVV08 protocol, a doctor's prescription is revealed by the doctor. To protect a doctor's prescription in these cases, one need to hide the link between a doctor and his prescription. We refer to prescribing-privacy as unlinkability of a doctor (indicated by a doctor identity) and his prescription. In case that the prescription is revealed in the doctor process, it is not suitable to model the unlinkability as the equivalence of two processes, one in which a doctor prescribes $\mathtt{a}$, and one in which the doctor prescribes $\mathtt{b}$, because obviously when the prescription is revealed in the doctor process $P_{dr}\{\mathtt{a}/\mathtt{presc}\} \not\approx_{\ell} P_{dr}\{\mathtt{b}/\mathtt{presc}\}$. Intuitively, prescribing-privacy is modelled as: given two honest users $\mathtt{A}$ and $\mathtt{B}$ prescribing two prescriptions $\mathtt{a}$ and $\mathtt{b}$, when the adversary saw the two prescriptions $\mathtt{a}$ and $\mathtt{b}$, the adversary does not know which doctor ($\mathtt{A}$ or $\mathtt{B}$) prescribed which prescription.

Definition 3 (Prescribing-privacy). *Let EHP be an eHealth protocol. Let $R_{dr} = \nu \mathtt{Id}_{dr}.init_{dr}.!P_{dr}$ be the doctor role. The protocol EHP satisfies prescribing-privacy if*

$$\begin{aligned} &\mathcal{C}[init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{a}/presc\}) \mid \\ &\quad init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{b}/presc\})] \\ \approx_{\ell}\ &\mathcal{C}[init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{b}/presc\}) \mid \\ &\quad init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{a}/presc\})], \end{aligned}$$

where $\mathcal{C}$ is a context which models the honest participants; process $init_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$ represents the process $init_{dr}$ with a variable $\mathtt{Id}_{dr}$ replaced by a free name $\mathtt{A}$; process $main_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{a}/\mathtt{presc}\})$ denotes the process $main_{dr}$ with two free variables Id_{dr} and presc, replaced by free names $\mathtt{A}$ and $\mathtt{a}$, respectively; $\mathtt{B}$ is a free name representing an honest doctor; $\mathtt{b}$ is a free name representing a prescription.

3.2 Enforced prescribing-privacy

Enforced privacy properties have been studied in other domains, for example, receipt-freeness and coercion-resistance (coercion-resistance implies receipt-freeness) in voting [13, 14], receipt-freeness in online auction [15]. It is also required in eHealth, for instance, a pharmaceutical company may bribe doctors to favour their medicines. To prevent doctors being bribed to favour certain medicines, doctor's privacy should be enforced by eHealth protocols (enforced prescribing-privacy).

Before defining enforced prescribing-privacy, we need to briefly introduce two definitions proposed in [13]. First, the process of a bribed user, in which the user actively communicates with the adversary (publishing his information) and tries to prove to the adversary his privacy, is defined as follows:

Let P be a plain process and $\mathtt{chc}$ a channel name. $P^{\mathtt{chc}}$, the process that shares all of P's secrets, is defined as:

- $0^{\mathtt{chc}} \hat{=} 0$,
- $(P \mid Q)^{\mathtt{chc}} \hat{=} P^{\mathtt{chc}} \mid Q^{\mathtt{chc}}$,
- $(\nu n.P)^{\mathtt{chc}} \hat{=} \nu n.\mathsf{out}(ch, n).P^{\mathtt{chc}}$ when n is a name of base type,
- $(\nu n.P)^{\mathtt{chc}} \hat{=} \nu n.P^{\mathtt{chc}}$ otherwise,
- $(\mathsf{in}(u, x).P)^{\mathtt{chc}} \hat{=} \mathsf{in}(u, x).\mathsf{out}(ch, x).P^{\mathtt{chc}}$ when x is a variable of base type,
- $(\mathsf{in}(u, x).P)^{\mathtt{chc}} \hat{=} \mathsf{in}(u, x).P^{\mathtt{chc}}$ otherwise,
- $(\mathsf{out}(u, M).P)^{\mathtt{chc}} \hat{=} \mathsf{out}(u, M).P^{\mathtt{chc}}$,
- $(!P)^{\mathtt{chc}} \hat{=} !P^{\mathtt{chc}}$,

- $(\mathit{if}\ M =_E N\ \mathit{then}\ P\ \mathit{else}\ Q)^{\mathsf{chc}} \hat{=} \mathit{if}\ M =_E N\ \mathit{then}\ P^{\mathsf{chc}}\ \mathit{else}\ Q^{\mathsf{chc}}$.

Second, a process erasing the output on a channel is defined as follows: Let P be an extended process. $P^{\backslash \mathsf{out}(ch,\cdot)}$ is defined as $P^{\backslash \mathsf{out}(ch,\cdot)} := \nu ch.(P \mid !\mathsf{in}(ch, x))$.

Similar to receipt-freeness in voting, enforced prescribing-privacy is defined as the existence of a way for a bribed/coerced doctor to lie about his prescription, while the adversary cannot tell whether the doctor lied. It is modelled as the existence of a process P', in which the bribed/coerced doctor can lie about his prescription, while the adversary cannot distinguish P' from the process in which the doctor genuinely reveal all his secret information to the adversary. This intuition is modelled as two equivalences. In the first equivalence, left hand side is $C[P']$, right hand side is the coerced doctor behaviour P^{chc}. It represents that the adversary cannot distinguish the doctor behaviour in P' and P^{chc}. In the second equivalence, left hand side is a process which is process P' erasing the doctor's communication with the adversary, right hand side is a doctor process in which the doctor prescribed differently from the adversary's order. It represents that the behaviour of P' in his communication partner's view (left hand side) looks the same as an honest doctor process in which the doctor prescribed differently form the adversary's expecting.

Definition 4 (Enforced prescribing-privacy). *Let EHP be an eHealth protocol. Let $R_{dr} = \nu \mathtt{Id}_{dr}.\mathit{init}_{dr}.!P_{dr}$ be the doctor role. The protocol EHP satisfies enforced prescribing-privacy, if there exists a process P'_{dr} such that:*

$$\begin{aligned}
&\mathcal{C}[(\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid P'_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\})) \mid \\
&\quad (\mathit{init}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}, a/\mathit{presc}\}))] \\
\approx_\ell\ &\mathcal{C}[((\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\})^{\mathtt{chc}}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid (\mathit{main}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}, \mathtt{a}/\mathit{presc}\})^{\mathtt{chc}})) \mid \\
&\quad (\mathit{init}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}, \mathtt{b}/\mathit{presc}\}))], \\
&(\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid P'_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}^{\backslash \mathsf{out}(\mathtt{chc},\cdot)})) \\
\approx_\ell\ &(\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}, \mathtt{b}/\mathit{presc}\})),
\end{aligned}$$

where $\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid P'_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\})$ is a closed plain process. In the definition, $\mathcal{C}$ is a context which models the honest participants; Id_{dr} and presc are free variables; `A` *and* `B` *are free names, representing doctor identities known by the adversary;* `a` *and* `b` *are two free names, representing two different prescriptions;* `chc` *is a channel not appeared in any process.*

3.3 Independency of prescribing-privacy

eHelath systems involves more than two roles. Some of them is able to access to sensitive data. However, not every role can be trusted, for example, pharmacists may be bribed by the adversary [10]. The untrusted role may reveal his information to the adversary such that privacy of other roles is broken. For instance, in eHealth, pharmacists may have sensitive data which can be revealed to help the adversary break a doctor's privacy. To prevent a party (not a doctor) help break a doctor's privacy, eHealth systems require that even if the party reveals his information, the adversary should not be able to break a doctor's privacy.

This property is named as independency of prescribing-privacy, which is defined as follows.

Definition 5 (Independency of prescribing-privacy). *Let EHP be an eHealth protocol. Let $R_{dr} = \nu \mathtt{Id}_{dr}.init_{dr}.!P_{dr}$ be the doctor role. Let R_i be a role in the protocol (R_i is not R_{dr}). The protocol EHP satisfies independency of prescribing-privacy if*

$$\begin{aligned}
&\mathcal{C}[!{R_i}^{\mathtt{chc}} \mid (init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{a}/presc\})) \mid \\
&\qquad\qquad (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{b}/presc\}))] \\
\approx_{\ell}\ &\mathcal{C}[!{R_i}^{\mathtt{chc}} \mid (init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{b}/presc\})) \mid \\
&\qquad\qquad (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{a}/presc\}))],
\end{aligned}$$

where $\mathcal{C}$ is a context which models the honest participants; Id_{dr} and presc are free variables; A *and* B *are free names, representing doctor identities known by the adversary;* a *is a free name, representing a prescription;* chc *is a channel not appeared in any process.*

3.4 Independency of enforced prescribing-privacy

Enforced prescribing-privacyassumes that a doctor reveals her information and independency of prescribing-privacy assumes a third party reveals his information to the adversary. It is nature to consider that the adversary is able to bribe/coerce both doctors and a third party to obtain more information. Since the adversary obtains more information, the doctor's privacy is potentially broken. To address this privacy problem, it requires a new privacy property: independency of enforced prescribing-privacy, which means a doctor's privacy is preserved even if the doctor and a third party reveal their information to the adversary. It can be considered as the combination of enforced prescribing-privacy and independency of prescribing-privacy.

Definition 6 (Independency of enforced prescribing-privacy). *Let EHP be an eHealth protocol. Let $R_{dr} = \nu \mathtt{Id}_{dr}.init_{dr}.!P_{dr}$ be the doctor role. Let R_i be a role in the protocol (R_i is not R_{dr}). The protocol EHP satisfies independency of enforced prescribing-privacyif there exists a process R'_{dr}, such that:*

$$\begin{aligned}
&\mathcal{C}[!(R_i)^{\mathtt{chc}} \mid ((init_{dr}\{\mathtt{A}/Id_{dr}\})^{\mathtt{chc}}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid P'_{dr}\{\mathtt{A}/Id_{dr}\})) \mid \\
&\qquad\qquad (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{a}/presc\}))] \\
\approx_{\ell}\ &\mathcal{C}[!(R_i)^{\mathtt{chc}} \mid ((init_{dr}\{\mathtt{A}/Id_{dr}\})^{\mathtt{chc}}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid (main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{a}/presc\})^{\mathtt{chc}})) \mid \\
&\qquad\qquad (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{b}/presc\}))], \\
&\qquad\qquad (init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid P'_{dr}\{\mathtt{A}/Id_{dr}\}^{\backslash \mathtt{out}(\mathtt{chc},\cdot)})) \\
&\qquad\quad \approx_{\ell} (init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{b}/presc\})),
\end{aligned}$$

and $(init_{dr}\{\mathtt{A}/Id_{dr}\})^{\mathtt{chc}}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid P'_{dr}\{\mathtt{A}/Id_{dr}\}))$ is a closed plain process. In the definition, $\mathcal{C}$ is a context which models the honest participants; Id_{dr} and presc are free variables; A *and* B *are free names, representing doctor identities known by the adversary;* a *and* b *arefree names, representing prescriptions;* chc *is a channel not appeared in any process.*

4 Anonymity and untraceabiltity

Anonymity and untraceability have been formally studied in the literature (e.g., [16–21]), which can be lifted to the eHealth domain.

4.1 Anonymity and strong anonymity

Anonymity is a property that protect users' identities. We model anonymity as indistinguishability of processes initiated by two different users.

Definition 7 (Doctor anonymity). *A well-formed eHealth protocol EHP satisfies doctor anonymity for a doctor* A *if there exists another doctor* B*, such that*

$$\mathcal{C}[\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}] \approx_\ell \mathcal{C}[\mathit{init}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\}.!P_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\}].$$

A stronger notion of anonymity is defined in [20], capturing the situation that the adversary cannot even find out whether a user (with identity A) has participated in a session of the protocol or not.

Definition 8 (Strong doctor anonymity [20]). *A well-formed eHealth protocol EHP satisfies strong doctor anonymity, if*

$$\mathit{EHP} \approx_\ell \nu\tilde{m}.\mathit{init}.\big(!R_1 \mid \ldots \mid !R_n \mid (\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\})\big).$$

Similarly, we can define anonymity and strong anonymity for patient and other roles in an eHealth protocol, by simply replacing the doctor role with a different role.

4.2 Untraceability and strong untraceability

Untraceability is a property preventing the adversary from tracing a user. It is defined as the adversary cannot tell whether two executions are initiated by the same user.

Definition 9 (Doctor untraceability). *A well-formed eHealth protocol EHP satisfies doctor untraceability if, for any two doctors* A *and* B $\neq$ A*,*

$$\begin{aligned} &\mathcal{C}[\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\})] \\ \approx_\ell\ &\mathcal{C}[(\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}) \mid (\mathit{init}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\}.P_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\})]. \end{aligned}$$

A stronger notion of untraceability is proposed in [20] that captures the adversary's inability to distinguish the situation in which one user executes the protocol multiple times from a situation in which no user executes the protocol more than once.

Definition 10 (Strong doctor untraceability [20]). *A well-formed eHealth protocol EHP satisfies strong doctor untraceability, if*

$$\mathit{EHP} \approx_\ell \nu\tilde{m}.\mathit{init}.\big(!R_1 \mid \ldots \mid !R_{i-1} \mid !R_{i+1} \mid !R_n \mid !(\nu\mathtt{Id}_{dr}.\mathit{init}_{dr}.P_{dr})\big).$$

Similarly, we can define untraceability and strong untraceability for patient and other roles in an eHealth protocol, by simply replacing the doctor role with a different role.

5 DLVV08 protocol

The DLVV08 protocol works as follows: a doctor prescripts medicine to a patient; the patient obtains medicine from a pharmacist according to the prescription; the pharmacist forwards prescriptions to the medicine prescription administrator (MPA), the administrator checks the prescriptions and refunds the pharmacist; the medicine administrator sends invoices to the patient's health insurance institute (HII) and get refunded.

5.1 Cryptographic primitives

To ensure users' security and privacy, the DLVV08 protocol employees several special cryptographic primitives, for instance, bit-commitments, zero-knowledge proofs, digital credentials (for anonymous authentication), signed proofs of knowledge, and verifiable encryptions.

Bit-commitments. The bit-commitments scheme consists of two phases, committing phase and opening phase. On the committing phase, a message sender makes a commitment on a message, which can be considered as putting the message into a box, and sending the box to the receiver. Later in the opening phase, the sender sends the key of the box to the receiver. The receiver opens the box and obtains the message.

Zero-knowledge proofs. A zero-knowledge proof is a cryptographic scheme which can be used for one party (prover) to prove to another party (verifier) that a statement is true, without leaking secret information of the prover. A zero-knowledge proof scheme can be interactive or non-interactive. We consider the non-interactive zero-knowledge proofs in this paper.

Digital credentials. A digital credential is like a certificate, which can be used to prove that the owner qualifies some requirements. Unlike some paper certificates such as passport which gives out the owner's identity, a digital credential could be used to authenticate the owner anonymously. For example, a digital credential can be used to prove that a driver is old enough to drive without showing the age of the driver.

Anonymous authentication. Anonymous authentication is a scheme for authenticating a user anonymously. In the scheme, a user's digital credential is used as the public key in the public key authentication structure. A verifier can check whether a message is signed correctly by the prover, while the verifier cannot identify the prover. Thus, this ensures anonymous authentication. The procedure of an anonymous authentication is actually a zero-knowledge proof, with the digital credential being the public information of the prover.

Verifiable encryptions. A verifiable encryption is a zero-knowledge proof as well. A prover encrypts a message, and uses zero-knowledge proofs to prove that the encrypted message satisfies some properties without showing the original message.

Signed proofs of knowledge. Signed proofs of knowledge is using proofs of knowledge as a digital signature scheme (for details see [22]). Intuitively, a prover signs a message using some secret information, which can be considered as a secret signing key. And the prover uses proofs of knowledge to convince the verifier that he has the secret signing key corresponding to the public key.

5.2 Settings

Roles in the DLVV08 protocol are equipped with initial knowledge. A doctor has an anonymous doctor credential. A patient has an anonymous patient credential. Doctor credentials and patient credentials are issued by trusted authorities (medical certification authority and central government-approved certification authority, respectively). Pharmacists, MPA, and HII are public entities, each of which has an authorised public key certificate issued by trusted authorities (government-approved certification organisations). Besides of credentials, a doctor has an identifier (doctor identity), a doctor pseudonym; a patient has an identifier (patient identity), a social security status, a health expense account maintained by his HII, and a patient pseudonym; a pharmacist has an identifier (pharmacist identity) and a corresponding MPA; an MPA have an identifier (MPA identity); and an HII has an identifier (HII identity).

5.3 Description of the DLVV08 protocol

The DLVV08 protocol consists of four sub-protocols: doctor-patient sub-protocol, patient-pharmacist sub-protocol, pharmacist-MPA sub-protocol, and MPA-HII sub-protocol.

Doctor-patient sub-protocol The doctor authenticates himself to a patient using the authorised doctor credential. The patient verifies the doctor credential. If the verification passes, the patient authenticates himself to the doctor using the patient credential, sends the patient bit-commitments on the patient's identity to the doctor, and proves to the doctor that the patient's identity used in the patient credential is the same as in the patient bit-commitments. After verifying the patient credential, the doctor generates a prescription, computes a prescription identity, computes the doctor bit-commitments. Then the doctor combines these computed messages with the received patient bit-commitments; signs these messages using a signed proof of knowledge, which proves that the doctor's pseudonym used in the doctor credential is the same as in the doctor bit-commitments. Together with the proof, the doctor sends the open information of the doctor bit-commitments.

Patient-Pharmacist sub-protocol The pharmacist authenticates himself to the patient. The patient verifies the authentication and obtains, from the authentication, the pharmacist's identity and the pharmacist's MPA. Then the patient anonymously authenticates himself to the pharmacist, and proves his social security status. Next, the patient computes verifiable encryptions vc_1, vc_2, vc_3, vc'_3, vc_4, vc_5, where

- vc_1 encrypts the patient's HII using the MPA's public key and proves that the HII encrypted in vc_1 is the same as the one in the patient's credential.
- vc_2 encrypts the doctor's pseudonym using the MPA's public key and proves that the doctor's pseudonym encrypted in vc_2 is the same as the one in the doctor commitment embedded in the prescription.
- vc_3 encrypts the patient's pseudonym using the public safety organisation's public key and proves that the pseudonym encrypted in vc_3 is the same as the one in the patient's commitment.
- vc'_3 encrypts the patient's HII using the social security organisation's public key and proves that the content encrypted in vc'_3 is the same as the HII in the patient's credential.
- vc_4 encrypts the patient's pseudonym using the MPA's public key and proves that the patient's pseudonym encrypted in vc_4 is the same as the one in the patient's credential.
- vc_5 encrypts the patient's pseudonym using his HII's public key and proves that the patient's pseudonym encrypted in vc_5 is the same as the one in the patient's credential.
- c_5 encrypts vc_5 using the MPA's public key.

The patient sends the received prescription to the pharmacist and proves to the pharmacist that the patient's identity in the prescription is the same as in the patient credential. The patient sends $vc_1, vc_2, vc_3, vc'_3, vc_4, c_5$ as well. The pharmacist verifies the correctness of all the received messages. If every message is correctly formated, the pharmacist charges the patient, and delivers the medicine. Then the pharmacist generates an invoice and sends it to the patient. The patient computes a receipt *ReceiptAck*: signing a message (consists of the prescription identity, the pharmacist's identity, vc_1, vc_2, vc_3, vc'_3, vc_4, vc_5) using a signed proof of knowledge and proving that he knows the patient credential. This receipt proves that the patient has received his medicine. The pharmacist verifies the correctness of the receipt.

Pharmacist-MPA sub-protocol The pharmacist and the MPA first authenticate each other using public key authentication. Then the pharmacist sends the received prescription and the receipt *ReceiptAck*, together with vc_1, vc_2, vc_3, vc'_3, vc_4, c_5, to the MPA. The MPA verifies correctness of the received information. Then, the MPA decrypts vc_1, vc_2, vc_4 and c_5, which provide the patient's HII, the doctor's pseudonym, the patient's pseudonym, and vc_5.

MPA-HII sub-protocol The MPA and the patient's HII first authenticate each other using public key authentication. Then the MPA sends the receipt *ReceiptAck* to the patient's HII as well as the verifiable encryption vc_5 which encrypts the patient's pseudonym with the patient's HII's public key. The patient's HII checks the correctness of *ReceiptAck*, decrypts vc_5 and obtains the patient's pseudonym. From the patient pseudonym, the HII obtains the identity of the patient; then updates the patient's account and pays the MPA. The MPA pays the pharmacist when he receives the payment.

Notice that the description of the protocol is slightly different from the original protocol in [10]. We do not care about the revocability, reimbursement and statistics property, therefore, we do not care about two roles mentioned in the paper: public safety organisation and social security organisation.

5.4 Ambiguities

The DLVV08 protocol leaves the following things open:

a1 whether a zero-knowledge proof is transferable;
a2 whether the encryption is probabilistic;
a3 whether a patient/doctor use a fresh identity and pseudonym in each session;
a4 whether a credential is fresh in each session;
a5 what a patient's social security status is and how it changes;
a6 how many HIIs and whether a patient's HII is changeable;
a7 whether a patient/doctor can obtain a credential by demanding;
a8 which kind of communicating channels are used;
a9 how a patient's health expense account changes;
a10 whether a pharmacist has a fixed MPA.

To discover security problems, we make the following assumptions:

s1 the zero-knowledge proofs used are non-interactive and transferable;
s2 the encryption is not probabilistic;
s3 a patient/doctor uses the same identity and pseudonym in every session;
s4 a patient/doctor has the same credential in every session and no one tries to misuse his credential;
s5 a patient's social security status is the same in every session;
s6 there are many HIIs, different patients may have different HIIs and a patient's HII is not changeable;
s7 a patient/doctor's credential can be obtained by demanding;
s8 the communicating channels are public;
s9 each patient has one health expense account and the account does not change;
s10 a pharmacist can communicate with any MPA as he likes.

5.5 Claimed privacy properties

The DLVV08 protocol is claimed to satisfy the following privacy properties:

- Secrecy of patient and doctor information: No other party should be able to know a patient or a doctor's information, unless the information is intended to be revealed in the protocol.

- Authentication: All parties should properly authenticate each other.
- prescribing-privacy: The protocol protects a doctor's prescription behaviour.
- enforced prescribing-privacy: The protocol prevents bribery between doctors and pharmaceutical companies.
- independency of prescribing-privacy: Pharmacists should not be able to provide evidence to pharmaceutical companies about doctors' prescription.
- patient anonymity: No party should be able to determine a patient's identity.
- patient untraceability: Prescriptions issued to the same patient should not be linkable to each other.

6 Modelling the DLVV08 protocol

We model the DLVV08 protocol in the applied pi calculus. Since the description of the protocol is not clear in some details, before modelling the protocol, a few ambiguities need to be settled. Next we explain the modelling of a few cryptographic primitives, since security and privacy properties rely heavily on these cryptographic primitives in the protocol. Then, we illustrate the modelling of the protocol.

6.1 Modelling cryptographic primitives

The cryptographic primitives are modelled in the applied pi calculus using functions and equational theory. All functions and equational theory are shown in Appendix A.

Bit-commitments. The bit-commitments scheme is modelled as two functions: function commit, modelling the committing phase, and function open, modelling the opening phase. Function commit creates a commitment with two parameters: a message m and a random number r. A commitment can only be opened with the correct opening information r, thus reveals the message m.

$$\begin{array}{rl} \textit{fun} & \mathsf{commit}/2. \\ \textit{reduc} & \mathsf{open}(\mathsf{commit}(m, r), r) = m. \end{array}$$

Note that key word *fun* is used to declaim function in Proverif and key word *reduc* is used to declaim the equational theory in ProVerif.

Zero-knowledge proofs. Non-interactive zero-knowledge proofs can be modelled as function $\mathsf{zk}(\mathit{secrets}, \mathit{pub\_info})$ (a function with two parameters: a tuple of secrete information *secrets*, and a tuple of public information *pub_info*) inspired by [23][1]. The verifying information and the secret information satisfies a relation. Since the secret information is only known by the prover, only the prover can construct the zero-knowledge proof. To verify a zero-knowledge proof is to

[1] We define each zero-knowledge specifically, comparing to that in [23], because there is limited number of zero-knowledge proofs

check whether the relation between the secret formation and the verifying formation is satisfied. The verification of a zero-knowledge proof is modelled as function $\mathsf{VerifyZk}(\mathsf{zk}(\mathit{secrets}, \mathit{pub\_info}), \mathit{verif\_info})$, in which two parameters are: a zero-knowledge proof to be verified $\mathsf{zk}(\mathit{secrets}, \mathit{pub\_info})$ and the verification information $\mathit{verif\_info}$.

We specify each verification rule in this paper. Since the $\mathit{pub\_info}$ and $\mathit{verif\_info}$ happens to be the same in all the zero-knowledge proofs verifications in this paper, the generic structure of verification rule is as

$$\mathsf{VerifyZk}(\mathsf{zk}(x, \mathsf{f}(x, y)), \mathsf{f}(x, y)) = \mathsf{true},$$

where x denotes secret information and y denotes public information.

For example, let c be a secret used to compute both $a = \mathsf{f}(c)$ and $b = \mathsf{g}(c)$. To prove that the same value was used in both computations, the prover constructs the zero-knowledge proof $\mathsf{zk}(c, (a, b))$. To verify a given proof z, the verifier executes $\mathsf{VerifyZkEx}(z, \mathsf{getpublic}(z))$, defined as:

$$\mathsf{VerifyZkEx}(\mathsf{zk}(x, (\mathsf{f}(x), \mathsf{g}(x)),\ (\mathsf{f}(x), \mathsf{g}(x)))\ ) = \mathsf{true}.$$

When the proof z is constructed as $\mathsf{zk}(x, (\mathsf{f}(x), \mathsf{g}(x)))$, the proof is verified. Otherwise the verification fails. Since x is secret and cannot be derived from a and b, the adversary cannot forge a proof which satisfies the verification.

$$\begin{array}{rl} \mathit{fun} & \mathsf{zk}/2. \\ \mathit{fun} & \mathsf{true}/0. \\ \mathit{reduc} & \mathsf{VerifyZk}(\mathsf{zk}(x, \mathsf{f}(x, y)), \mathsf{f}(x, y)) = \mathsf{true}. \\ \mathit{reduc} & \mathsf{getpublic}(\mathsf{zk}(x, y)) = y. \end{array}$$

Digital credentials. A digital credential is issued by trusted authorities. We assume the procedure of issuing a credential is perfect, which means that the adversary cannot forge a credential nor obtain one by impersonation. We model digital credentials as a private function (declaimed by key word *private fun* in ProVerif) which is only usable by honest users. In the DLVV08 protocol, a credential can have several attributes; we model these as parameters of the credential function.

$$\begin{array}{l} \mathit{private\ fun}\ \mathsf{drcred}/2. \\ \mathit{private\ fun}\ \mathsf{ptcred}/5. \end{array}$$

There are two credentials in the DLVV08 protocol: a doctor credential which is modelled as $\mathit{Cred}_{dr} = \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})$, and a patient credential which is modelled as $\mathit{Cred}_{pt} = \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})$.

Anonymous authentication. The procedure of anonymous authentication is a zero-knowledge proof using the digital credential as public information. The anonymous authentication of a doctor is modelled as

$$\mathit{Auth}_{dr} = \mathsf{zk}((y, z), \mathsf{drcred}(y, z)),$$

and the verification of the authentication is modelled as $\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{dr}}(Auth_{dr}, \mathsf{drcred}(y, z))$. The equational theory for the verification is

$$\textit{reduc}\ \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{dr}}(\mathsf{zk}((y, z), \mathsf{drcred}(y, z)), \mathsf{drcred}(y, z)) = \mathsf{true}.$$

The verification implies that the creator of the authentication is a doctor, because only doctors can use the function drcred, and thus create a valid proof. The adversary can observe a credential $\mathsf{drcred}(y, z)$, but does not know secrets y, z, and thus cannot forge a valid zero-knowledge proof. If the adversary forges a zero-knowledge proof with fake secret information y' and z', the fake zero-knowledge proof will not pass verification. For the same reason, a validated proof proves that the credential belongs to the creator of the zero-knowledge proof.

Similarly, an anonymous authentication of a patient is modelled as

$$\begin{aligned} Auth_{pt} = \mathsf{zk}(\ &(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})), \end{aligned}$$

and the verification rule is modelled as

$$\begin{aligned} \textit{reduc}\ \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(\ &\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &\quad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})),\\ &\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})) = \mathsf{true}. \end{aligned}$$

Verifiable encryptions. A verifiable encryption is modelled as a zero-knowledge proof. The encryption is embedded in the zero-knowledge proof as public function. The receiver can obtain the cipher text from the proof. For example, a patient wants to prove that he encrypted a secrete s using a public key k to a pharmacist, while the pharmacist does not know the corresponding secrete key for k. The pharmacist cannot open the cipher text to test whether it uses the public key k to encrypt. However, the zero-knowledge proof can prove that the cipher text is encrypted using k, while not revealing s.

The general structure of the verification of a verifiable encryption is

$$\mathsf{VerifyVenc}(\mathsf{zk}(\textit{secrets}, (\textit{pub\_info}, \textit{cipher})), \textit{verif\_info}) = \mathsf{true},$$

where *secrets* is private information, *pub_info* and *micipher* consist public information, *verif_info* is the verification information.

Signed proofs of knowledge. A signed proof of knowledge is a scheme which signs a message, and proves a property of the signer. For the DLVV08 protocol, this proof only concerns equality of attributes of credentials and commitments (e.g. the identify of this credential is the same as the identity of that commitment).

To verify a signed proof of knowledge, the verifier must know which credentials/commitments are considered. Hence, this information must be obtainable from the proof, and thus is included in the model. In general, a signed proof of knowledge is modelled as function

$$\mathsf{spk}(\textit{secrets}, \textit{pub\_info}, \textit{msg}),$$

which models a signature using private value(s) *secrets* on the message *msg*, with public information *pub_info* as settings.

What knowledge is proven, depends on the specific instance of the proof and is captured by the verification functions for the specific proofs. For example, to prove that a user knows a) all fields of a (simplified) credential, b) all fields of a commitment to an identity, and c) that the commitment concerns the same identity as the commitment, he generates the following proof:

$$\begin{aligned}\mathsf{spk}(\ &(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, r_{pt}),\\ &(\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}), \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})),\\ &msg).\end{aligned}$$

These proofs are verified by checking that the signature is correct, given the signed message and the verification information. Generically, this is modelled as $\mathsf{VerifySpk}\left(\mathsf{spk}\left(x, \mathsf{f}(x,y), m\right), \mathsf{f}(x,y), m\right) = \mathsf{true}$, although the specific verification function depends on the specific proof to be verified. E.g., the above example proof can be verified as follows:

$$\begin{aligned}\textit{reduc}\ \mathsf{VerifySpk}(\ \mathsf{spk}(\ &(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, r_{pt}),\\ &(\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}), \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})),\\ &msg\ ),\\ (\ &\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}), \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})\ ),\\ &msg\\ ) &= \mathsf{true}.\end{aligned}$$

Consequently, the spk-related functions are generically modelled as follows.

$$\begin{aligned}\textit{fun}\ \ &\mathsf{spk}/3.\\ \textit{reduc}\ \ &\mathsf{VerifySpk}(\mathsf{spk}(x, \mathsf{f}(x,y), m), \mathsf{f}(x,y), m) = \mathsf{true}.\\ \textit{reduc}\ \ &\mathsf{getSpkVinfo}(\mathsf{spk}(x,y,z)) = y.\\ \textit{reduc}\ \ &\mathsf{getSpkMsg}(\mathsf{spk}(x,y,z)) = z.\end{aligned}$$

6.2 Modelling the DLVV08 protocol

Settings Below lists the raw knowledge of each role.

- A doctor is initialised with: an identity $\mathtt{Id}_{dr}$, a pseudonym $\mathtt{Pnym}_{dr}$, and a credential $\textit{Cred}_{dr} = \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})$.
- A patient is initialised with: an identity $\mathtt{Id}_{pt}$, a pseudonym $\mathtt{Pnym}_{pt}$, an HII $\mathtt{Hii}$, a social security status $\mathtt{Sss}$, a health expense account $\mathtt{Acc}$ and a credential
  $\textit{Cred}_{pt} = \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})$.
- A pharmacist is initialised with: a secret key $\mathtt{sk}_{ph}$, a public key $\mathtt{pk}_{ph}$, and an identity $\mathtt{Id}_{ph}$.
- An MPA is initialised with: a secret key $\mathtt{sk}_{mpa}$, a public key $\mathtt{pk}_{mpa}$, and an identity Id_{mpa}.
- An HII is initialised with: a secret key $\mathtt{sk}_{hii}$, a public key $\mathtt{pk}_{hii}$, and an identity $\mathtt{Id}_{hii}$.

Besides, there is a public key of social security organisation $\mathtt{pk}_{sso}$ as a global public information.

The adversary initially knows the set of doctor identities, the set of patient identities,the set of pharmacist identities, the set of MPA identities, the set of HII identities and all public keys: $\mathtt{pk}_{ph}$ of each pharmacist, $\mathtt{pk}_{mpa}$ of each MPA, $\mathtt{pk}_{hii}$ of each HII and $\mathtt{pk}_{sso}$ of social security organisation.

Modelling Doctor-Patient sub-protocol This sub-protocol is to prescribe medicines for a patient. It contains two steps: first, doctor and patient anonymously authentications each other, second, the doctor sends prescription to the patient. The communication in the doctor-patient sub-protocol are as in Figure 6.2.

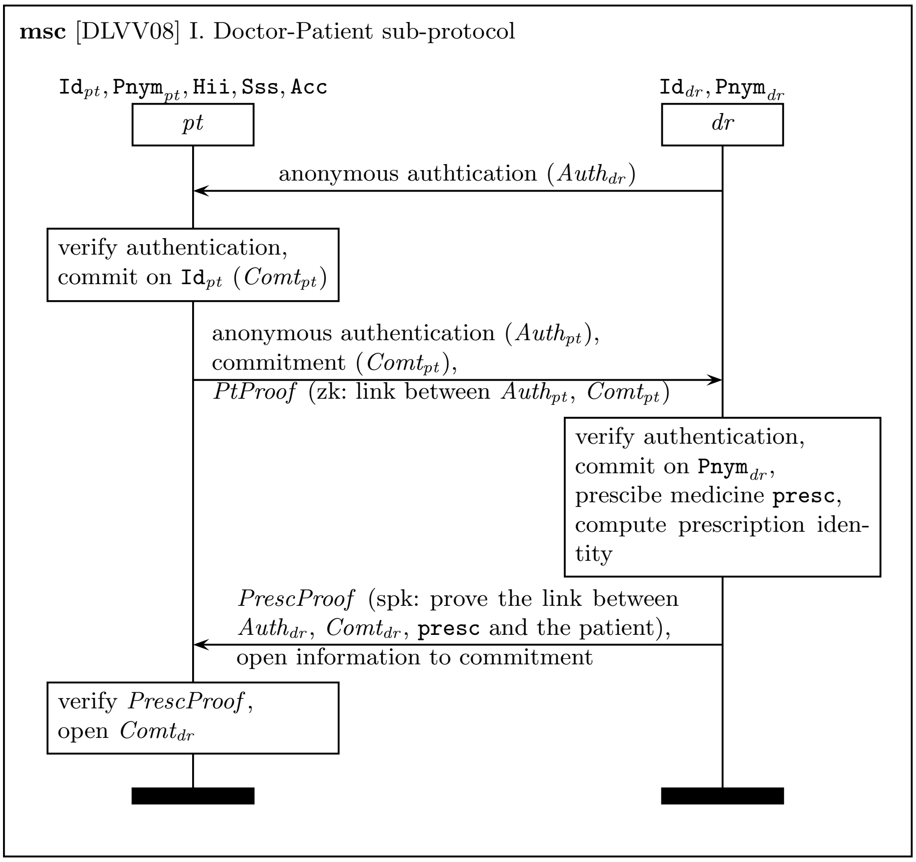

msc [DLVV08] I. Doctor-Patient sub-protocol
$\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}$
pt
$\mathtt{Id}_{dr}, \mathtt{Pnym}_{dr}$
dr
anonymous authtication ($Auth_{dr}$)
verify authentication, commit on $\mathtt{Id}_{pt}$ ($Comt_{pt}$)
anonymous authentication ($Auth_{pt}$), commitment ($Comt_{pt}$), *PtProof* (zk: link between $Auth_{pt}$, $Comt_{pt}$)
verify authentication, commit on $\mathtt{Pnym}_{dr}$, prescibe medicine $\mathtt{presc}$, compute prescription identity
PrescProof (spk: prove the link between $Auth_{dr}$, $Comt_{dr}$, $\mathtt{presc}$ and the patient), open information to commitment
verify *PrescProof*, open $Comt_{dr}$

Fig. 2. Doctor-Patient sub-protocol

Each step in Figure 6.2 is modelled as follows:

1. The doctor anonymously authenticates ($Auth_{dr}$) himself to the patient using his credential $Cred_{dr} = \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})$.

   $$Auth_{dr} = \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), Cred_{dr})$$

2. The patient reads in the doctor authentication rcv_Auth_{dr}, obtains the doctor credential

   $$c\_Cred_{dr} = \mathsf{getpublic}(rcv\_Auth_{dr})$$

   and verifies the authentication as follows:

   $$\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{dr}}(rcv\_Auth_{dr}, c\_Cred_{dr}).$$

   If the verification succeeds, according to the rule

   $$\begin{aligned} reduc\ \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{dr}}(\ & \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})), \\ & \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})) = \mathsf{true}, \end{aligned}$$

   the patient
   - anonymously authenticates ($Auth_{pt}$) himself to the doctor using his credential $Cred_{pt} = \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})$,

     $$Auth_{pt} = \mathsf{zk}(\ (\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), Cred_{pt});$$

   - generates a nonce r_{pt} and sends the patient bit-commitments,

     $$Comt_{pt} = \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt});$$

   - generates a proof $PtProof$ which proves that the patient identity used in the patient credential is the same as in the patient bit-commitments, thus links the patient bit-commitments and the patient credential. The proof is modelled as:

     $$\begin{aligned} PtProof = \mathsf{zk}(\ & (\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\ & (Comt_{pt}, Cred_{pt})). \end{aligned}$$

3. The doctor reads in the patient authentication rcv_Auth_{pt} and the proof $rcv\_PtProof$, obtains the patient credential from the patient authentication

   $$c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt}),$$

   obtains the patient commitment and the patient credential from the patient proof and tests whether the credential matches the one embeded in the patient authentication

   $$(c\_Comt_{pt}, =c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)^2,$$

[2] The $(=B) = \mathsf{f}(C)$ notation tests whether $\mathsf{f}(C)$ (applying function f on term C) matches B and aborts if not.

then verifies the two proofs as follows: [3]:

$$\begin{array}{l}\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt})\\ \mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt}))\end{array}$$

If the verification succeeds, according to the following rules respectively,

$$\begin{array}{lll} reduc & \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}} & (\mathsf{zk}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ & & \quad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})),\\ & & \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})) = \mathsf{true}.\\ reduc & \mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}} & (\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ & & (\mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}),\\ & & \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}))),\\ & & \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}),\\ & & \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})) = \mathsf{true}. \end{array}$$

the doctor generates a prescription[4] `presc`, generates a nonce $\mathtt{r}_{dr}$, computes a commitment

$$Comt_{dr} = \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}),$$

and a prescription identity

$$PrescriptID = \mathsf{hash}(\mathtt{presc}, c\_Comt_{pt}, Comt_{dr}).$$

Next, the doctor signs the message ($\mathtt{presc}$, $PrescriptID$, $Comt_{dr}$, c_Comt_{pt}) using a signed proof of knowledge *PrescProof*. This proof proves that the pseudonym used in the credential $Cred_{dr}$ is the same as in the commitment $Comt_{dr}$, thus linking the prescription to the credential.

$$\begin{array}{ll} PrescProof = \mathsf{spk}(& (\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),\\ & (Comt_{dr}, Cred_{dr}),\\ & (\mathtt{presc}, PrescriptID, Comt_{dr}, c\_Comt_{pt})). \end{array}$$

The doctor sends to the patient the message *PrescProof* together with the open information of the doctor commitment $\mathtt{r}_{dr}$.

4. The patient reads in the prescription as *rcv_PrescProof*, obtains messages: the prescription *c_presc*, prescription identity *c_PrescriptID*, doctor commitment c_Comt_{dr}, and tests the patient commitment signed in the receiving message

$$\begin{array}{r}(c\_presc, c\_PrescriptID, c\_Comt_{dr}, = \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}))\\ = \mathsf{getSpkMsg}(rcv\_PrescProof),\end{array}$$

verifies the proof *rcv_PrescProof* as follows:

$$\begin{array}{r}\mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(rcv\_PrescProof, (c\_Cred_{dr}, c\_presc, c\_PrescriptID,\\ c\_Comt_{dr}, \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}))),\end{array}$$

[3] Note that the equal notion in $(c\_Comt_{pt}, = c\_Cred_{pt})$ is a test, if the second field of the message obtaining from $\mathsf{getpublic}(rcv\_PtProof)$ is the same as c_Cred_{pt}, then continue the process, otherwise, stop.

[4] Notice that a medical examination of the patient is not part of the DLVV08 protocol.

according to the following rule,

$$\begin{aligned}\textit{reduc}\ \mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(\ & \mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),\\ &\quad (\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),\\ &\quad (\mathtt{presc}, \mathit{PrescriptID}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}),\\ &\quad \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}))),\\ & \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathtt{presc}, \mathit{PrescriptID},\\ & \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}),\\ & \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})) = \mathsf{true}.\end{aligned} \tag{R1}$$

if the verification succeeds, the patient obtains c_Pnym_{dr} by opening the doctor commitment

$$c\_Pnym_{dr} = \mathsf{open}(c\_Comt_{dr}, rcv\_r_{dr}).$$

From above, we can model the behaviour of doctor and patient, as P_{dr} and $P_{pt}\_p_1$ shown in Figure 3 and Figure 4, respectively. In applied pi, the doctor-patient sub-protocol is modelled as a doctor process and a patient process running in parallel, $P_{dr} \mid P_{pt}\_p_1$.

```
let $P_{dr}$ =
        $\mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})));$
        $\mathsf{in}(\mathtt{ch}, (rcv\_Auth_{pt}, rcv\_PtProof));$
        let $c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt})$ in
        let $(c\_Comt_{pt}, = c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt}) = \mathtt{true}$ then
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt})) = \mathtt{true}$ then
        $\nu\mathtt{presc};$
        $\nu\mathtt{r}_{dr};$
        let $PrescriptID = \mathsf{hash}(\mathtt{presc}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))$ in
        $\mathsf{out}(\mathtt{ch}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),$
                $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),$
                $(\mathtt{presc}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
            $\mathtt{r}_{dr})).$
```

Fig. 3. The doctor process P_{dr}.

Modelling Patient-Pharmacist sub-protocol This sub-process is used for a patient to obtain medicines from a pharmacist. It contains mainly five steps: first the patient and the pharmacist authenticate each other, second, the patient sends prescription to the pharmacist, third, the pharmacist sends invoice to the patient, fourth, the patient sends back a receipt to the pharmacist. The communication of this sub-protocol is shown in Figure 6.2[5].

[5] The dashed arrows are out of the scope of the protocol, thus, the message exchanging are not modelled.

```
let $P_{pt}\_p_1 =$
        in(ch, $rcv\_Auth_{dr}$);
        let $c\_Cred_{dr}$ = getpublic($rcv\_Auth_{dr}$) in
        if Vfy-zk$_{\mathsf{Auth}_{dr}}$($rcv\_Auth_{dr}$, $c\_Cred_{dr}$) = true then
        $\nu r_{pt}$;
        out(ch, (zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc)),
                 zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (commit(Id$_{pt}$, $r_{pt}$),
                     ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc)))));
        in(ch, ($rcv\_PrescProof$, $rcv\_r_{dr}$));
        let ($c\_presc$, $c\_PrescriptID$, $c\_Comt_{dr}$, = commit(Id$_{pt}$, $r_{pt}$))
           = getSpkMsg($rcv\_PrescProof$) in
        if Vfy-spk$_{\mathsf{PrescProof}}$($rcv\_PrescProof$, ($c\_Cred_{dr}$, $c\_presc$,
                               $c\_PrescriptID$, $c\_Comt_{dr}$, commit(Id$_{pt}$, $r_{pt}$))) = true then
        let $c\_Pnym_{dr}$ = open($c\_Comt_{dr}$, $rcv\_r_{dr}$) in 0.
```

Fig. 4. The patient process P_{pt} in Doctor-Patient sub-protocol.

Each step in Figure 6.2 is modelled as follows:

1. A pharmacist authenticates to the patient using public key authentication

$$\mathsf{sign}((\mathtt{Id}_{ph}, c_{ph}\_Id_{mpa}), \mathsf{sk}_{ph}).$$

   Note that the pharmacist does not authenticate anonymously. The authentication is modelled as signing a message using the pharmacist's secret key. Since the patient can obtain the pharmacists's MPA identity from the pharmacist authentication. We put the pharmacist's MPA identity and the pharmacist identity as messages.
2. The patient reads in the pharmacist's authentication rcv_Auth_{ph}, verifies the authentication using the public key of pharmacist $rcv_{pt}\_pk_{ph}$,

$$\mathsf{Vfy\text{-}sign}(rcv\_Auth_{ph}, rcv_{pt}\_pk_{ph}),$$

   according to the following rule,

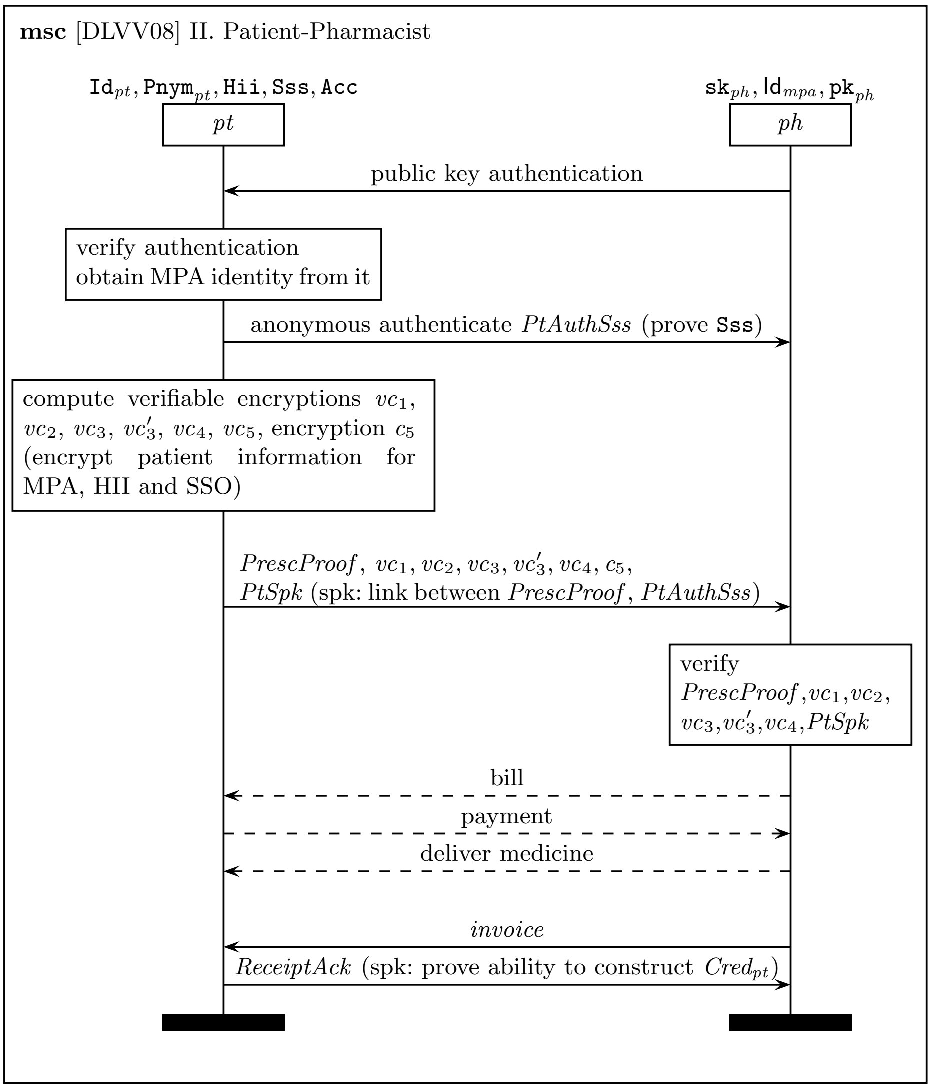
$$\textit{reduc}\ \mathsf{Vfy\text{-}sign}(\mathsf{sign}(x, y), \mathsf{pk}(y)) = \mathsf{true}. \qquad \text{(R2)}$$

   if the verification succeeds, the pharmacist obtains the pharmacist's identity from the authentication.

$$c_{pt}\_Id_{ph} = \mathsf{getsignmsg}(rcv\_Auth_{ph}, rcv_{pt}\_pk_{ph})$$

   Then the patient anonymously authenticates himself to the pharmacist, and proves his social security status using the proof $PtAuthSss$.

$$PtAuthSss = \mathsf{zk}(\ (\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\ (\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \mathtt{Sss}))$$


msc [DLVV08] II. Patient-Pharmacist
$\texttt{Id}_{pt}, \texttt{Pnym}_{pt}, \texttt{Hii}, \texttt{Sss}, \texttt{Acc}$
pt
$\texttt{sk}_{ph}, \mathsf{Id}_{mpa}, \texttt{pk}_{ph}$
ph
public key authentication
verify authentication obtain MPA identity from it
anonymous authenticate *PtAuthSss* (prove $\texttt{Sss}$)
compute verifiable encryptions vc_1, vc_2, vc_3, vc_3', vc_4, vc_5, encryption c_5 (encrypt patient information for MPA, HII and SSO)
PrescProof, $vc_1, vc_2, vc_3, vc_3', vc_4, c_5$, *PtSpk* (spk: link between *PrescProof*, *PtAuthSss*)
verify *PrescProof*,vc_1,vc_2, vc_3,vc_3',vc_4,*PtSpk*
bill
payment
deliver medicine
invoice
ReceiptAck (spk: prove ability to construct $Cred_{pt}$)

The patient computes the following verifiable encryptions. These verifiable encryptions are used to inform the MPA the patient's HII, the doctor's pseudonym, the patient's pseudonym, inform the HII the patient's pseudonym and inform the social safety organisation the patient pseudonym and the patient HII. The verifiable encryptions prove that the message encrypted is the message should be encrypted, without showing the message to those who cannot open the cipher text.

$$\begin{aligned}
vc_1 &: (\mathsf{enc}(\mathtt{Hii}, \mathtt{pk}_{mpa}), \text{proof that this is the same } \mathtt{Hii} \text{ as in } \mathit{Cred}_{pt})\\
vc_2 &: (\mathsf{enc}(\mathit{c\_Pnym}_{dr}, \mathtt{pk}_{mpa}),\\
&\quad \text{proof that this is the same } \mathit{c\_Pnym}_{dr} \text{ as in } \mathit{rcv\_PrescProof})\\
vc_3 &: (\mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{sso}), \text{proof that this is the same } \mathtt{Pnym}_{pt} \text{ as in } \mathit{Comt}_{pt})\\
vc_3' &: (\mathsf{enc}(\mathtt{Hii}, \mathtt{pk}_{sso}), \text{proof that this is the same } \mathtt{Hii} \text{ as in } \mathit{Cred}_{pt})\\
vc_4 &: (\mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{mpa}), \text{proof that this is the same } \mathtt{Pnym}_{pt} \text{ as in } \mathit{Cred}_{pt})\\
vc_5 &: (\mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{hii}), \text{proof that this is the same } \mathtt{Pnym}_{pt} \text{ as in } \mathit{Cred}_{pt})\\
c_5 &: (\mathsf{enc}(vc_5, \mathtt{pk}_{mpa}))
\end{aligned}$$

The verifiable encryptions are modelled as zero-knowledge proofs:

$$\begin{aligned}
&\text{let } \mathit{ptSecrets} = (\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})\\
&\text{let } \mathit{Cred}_{pt} \quad = \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})
\end{aligned}$$

$$\begin{aligned}
vc_1 &= \mathsf{zk}(\mathit{ptSecrets}, (\mathit{Cred}_{pt}, \mathsf{enc}(\mathtt{Hii}, \mathtt{pk}_{mpa})))\\
vc_2 &= \mathsf{zk}((\mathit{c\_Pnym}_{dr}, \mathit{rcv\_r}_{dr}), (\mathit{rcv\_PrescProof}, \mathsf{enc}(\mathit{c\_Pnym}_{dr}, \mathtt{pk}_{mpa})))\\
vc_3 &= \mathsf{zk}(\mathit{ptSecrets}, (\mathit{Cred}_{pt}, \mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{sso})))\\
vc_3' &= \mathsf{zk}(\mathit{ptSecrets}, (\mathit{Cred}_{pt}, \mathsf{enc}(\mathtt{Hii}, \mathtt{pk}_{sso})))\\
vc_4 &= \mathsf{zk}(\mathit{ptSecrets}, (\mathit{Cred}_{pt}, \mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{mpa})))\\
vc_5 &= \mathsf{zk}(\mathit{ptSecrets}, (\mathit{Cred}_{pt}, \mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{hii})))\\
c_5 &= \mathsf{enc}(vc_5, \mathtt{pk}_{mpa})
\end{aligned}$$

The patient computes a signed proof of knowledge *PtSpk* which proves that the patient identity embedded in the prescription is the same as in his patient credential:

$$\begin{aligned}
\mathit{PtSpk} = \mathsf{spk}(\ &(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\
&(\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})),\\
&\mathtt{nonce})
\end{aligned}$$

The patient sends the prescription *rcv_PrescProof*, the signed proof *PtSpk*, and vc_1, vc_2, vc_3, vc_3', vc_4, c_5 to the pharmacist.

3. The pharmacist reads in messages: *rcv_PtAuthSss*, $\mathit{rcv}_{ph}\mathit{\_PrescProof}$, $\mathit{rcv}_{ph}\mathit{\_PtSpk}$, $\mathit{rcv\_vc}_1$, $\mathit{rcv\_vc}_2$, $\mathit{rcv\_vc}_3$, $\mathit{rcv\_vc}_3'$, $\mathit{rcv\_vc}_4$; verifies the correctness of them as follows:
   - *rcv_PtAuthSss* (the received anonymous authentication and proof of social security status),

$$\begin{aligned}
(c_{ph}\mathit{\_Cred}_{pt}, c_{ph}\mathit{\_Sss}) &= \mathsf{getpublic}(\mathit{rcv\_PtAuthSss})\\
\mathsf{Vfy\text{-}zk}_{\mathsf{PtAuthSss}}&(\mathit{rcv\_PtAuthSss}, (c_{ph}\mathit{\_Cred}_{pt}, c_{ph}\mathit{\_Sss}))
\end{aligned}$$

using the following rule

$$\begin{aligned}\textit{reduc}\ \mathsf{Vfy\text{-}zk}_{\mathsf{PtAuthSss}}(\ &\mathsf{zk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc}),\\ &\qquad(\mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc}),\mathtt{Sss}),\\ &\mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc}),\mathtt{Sss}) = \mathsf{true}.\end{aligned}$$

- $\mathit{rcv}_{ph}\_\mathit{PrescProof}$ (the prescription), using the rule $R1$,

$$\begin{aligned}&(c_{ph}\_\mathit{Comt}_{dr},\, c_{ph}\_\mathit{Cred}_{dr}) = \mathsf{getSpkVinfo}(\mathit{rcv}_{ph}\_\mathit{PrescProof})\\ &(c_{ph}\_\mathit{presc},\, c_{ph}\_\mathit{PrescriptID}, = c_{ph}\_\mathit{Comt}_{dr},\, c_{ph}\_\mathit{Comt}_{pt})\\ &\qquad\qquad = \mathsf{getSpkMsg}(\mathit{rcv}_{ph}\_\mathit{PrescProof})\\ &\mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(\mathit{rcv}_{ph}\_\mathit{PrescProof}, (c_{ph}\_\mathit{Cred}_{dr},\, c_{ph}\_\mathit{presc},\, c_{ph}\_\mathit{PrescriptID},\\ &\qquad\qquad c_{ph}\_\mathit{Comt}_{dr},\, c_{ph}\_\mathit{Comt}_{pt}))\end{aligned}$$

- $\mathit{rcv}_{ph}\_\mathit{PtSpk}$ (the proof that the patient identity in the patient credential is the same as in the patient bit-commitments),

$$\begin{aligned}&c\_\mathit{msg} = \mathsf{getSpkMsg}(\mathit{rcv}_{ph}\_\mathit{PtSpk})\\ &\mathsf{Vfy\text{-}spk}_{\mathsf{PtSpk}}(\mathit{rcv}_{ph}\_\mathit{PtSpk}, (c_{ph}\_\mathit{Cred}_{pt},\, c_{ph}\_\mathit{Comt}_{pt},\, c\_\mathit{msg}))\end{aligned}$$

according to the following rule:

$$\begin{aligned}\textit{reduc}\ \mathsf{Vfy\text{-}spk}_{\mathsf{PtSpk}}(\ &\mathsf{spk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},r_{pt}),\\ &\quad(\mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc}),\mathsf{commit}(\mathtt{Id}_{pt},r_{pt})),\\ &\quad\mathtt{nonce}),\\ &\mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc}),\\ &\mathsf{commit}(\mathtt{Id}_{pt},r_{pt}),\mathtt{nonce}) = \mathsf{true}.\end{aligned}$$

- $\mathit{rcv}\_\mathit{vc}_1, \mathit{rcv}\_\mathit{vc}_2, \mathit{rcv}\_\mathit{vc}_3, \mathit{rcv}\_\mathit{vc}'_3, \mathit{rcv}\_\mathit{vc}_4$ (verifiable encryptions),

$$\begin{aligned}(= c\_\mathit{Cred}_{pt},\, c\_\mathit{Enc}_1) &= \mathsf{getpublic}(\mathit{rcv}\_\mathit{vc}_1)\\ (= \mathit{rcv}_{ph}\_\mathit{PrescProof},\, c\_\mathit{Enc}_2) &= \mathsf{getpublic}(\mathit{rcv}\_\mathit{vc}_2)\\ (= c_{ph}\_\mathit{Cred}_{pt},\, c\_\mathit{Enc}_3) &= \mathsf{getpublic}(\mathit{rcv}\_\mathit{vc}_3)\\ (= c_{ph}\_\mathit{Cred}_{pt},\, c\_\mathit{Enc}'_3) &= \mathsf{getpublic}(\mathit{rcv}\_\mathit{vc}'_3)\\ (= c_{ph}\_\mathit{Cred}_{pt},\, c\_\mathit{Enc}_4) &= \mathsf{getpublic}(\mathit{rcv}\_\mathit{vc}_4)\end{aligned}$$

$$\begin{aligned}&\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(\mathit{rcv}\_\mathit{vc}_1, (c_{ph}\_\mathit{Cred}_{pt},\, c\_\mathit{Enc}_1, \mathtt{pk}_{mpa}))\\ &\mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(\mathit{rcv}\_\mathit{vc}_2, (\mathit{rcv}_{ph}\_\mathit{PrescProof},\, c\_\mathit{Enc}_2, \mathtt{pk}_{mpa}))\\ &\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(\mathit{rcv}\_\mathit{vc}3, (c_{ph}\_\mathit{Cred}_{pt},\, c\_\mathit{Enc}_3, \mathtt{pk}_{sso}))\\ &\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(\mathit{rcv}\_\mathit{vc}'_3, (c_{ph}\_\mathit{Cred}_{pt},\, c\_\mathit{Enc}'_3, \mathtt{pk}_{sso}))\\ &\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(\mathit{rcv}\_\mathit{vc}_4, (c_{ph}\_\mathit{Cred}_{pt},\, c\_\mathit{Enc}_4, \mathtt{pk}_{mpa}))\end{aligned}$$

⋆ Notice that, for simplicity, the verification functions of vc_1 and vc'_3 are merged into one function $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}$, since the reduction of these two functions share the same structure. Similarly, the verification functions of vc_3 and vc_4 are merged into $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}$.

according to the following rules:

$$\begin{aligned}\mathit{reduc}\ \mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(\ &\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &\quad(\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &\quad\ \mathsf{enc}(\mathtt{Hii}, pk_x))),\\ &\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &\mathsf{enc}(\mathtt{Hii}, pk_x), pk_x) = \mathsf{true}.\end{aligned} \tag{R3}$$

$$\begin{aligned}\mathit{reduc}\ \mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(\ &\mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}),\\ &\quad(\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),\\ &\qquad\quad(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),\\ &\qquad\quad(\mathtt{presc}, \mathit{PrescriptID},\\ &\qquad\quad\ \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c_{ph}\text{\_}\mathit{Comt}_{pt})),\\ &\quad\ \mathsf{enc}(\mathtt{Pnym}_{dr}, pk_x))),\\ &\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),\\ &\quad(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),\\ &\quad(\mathtt{presc}, \mathit{PrescriptID},\\ &\quad\ \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c_{ph}\text{\_}\mathit{Comt}_{pt})),\\ &\mathsf{enc}(\mathtt{Pnym}_{dr}, pk_x), pk_x) = \mathsf{true}.\end{aligned} \tag{R4}$$

$$\begin{aligned}\mathit{reduc}\ \mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(\ &\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &(\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \mathsf{enc}(\mathtt{Pnym}_{pt}, pk_x))),\\ &\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &\mathsf{enc}(\mathtt{Pnym}_{pt}, pk_x), pk_x) = \mathsf{true}.\end{aligned} \tag{R5}$$

If all the verifications succeed, the pharmacist charges the patient, and delivers the medicine. Payment and delivery are out of the protocol's scope. Then the pharmacist generates an invoice and sends it to the patient. The invoice is a function of prescription identity,

$$\mathit{invoice} = \mathsf{invoice}\mathit{PrescriptID}.$$

4. The patient computes a receipt: a signed proof of knowledge *ReceiptAck* which proves that he knows the patient credential, as follows:

$$\begin{aligned}\mathit{ReceiptAck} = \mathsf{spk}(\ &(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ &(c\text{\_}\mathit{PrescriptID}, c_{pt}\text{\_}\mathit{Id}_{ph}, vc_1, vc_2, vc_3, vc'_3, vc_4, c_5)).\end{aligned}$$

5. The pharmacist reads in the receipt *rcv_ReceiptAck* and verifies its correctness as

$$\begin{aligned}&\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(\ \mathit{rcv\_ReceiptAck}, (c_{ph}\text{\_}\mathit{Cred}_{pt},\\ &\quad(c_{ph}\text{\_}\mathit{PrescriptID}, \mathtt{Id}_{ph},\\ &\quad\ \mathit{rcv\_vc}_1, \mathit{rcv\_vc}_2, \mathit{rcv\_vc}_3, \mathit{rcv\_vc}'_3, \mathit{rcv\_vc}_4, \mathit{rcv\_c}_5))),\end{aligned}$$

according to the rule:

$$\begin{aligned}\textit{reduc}\ \mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(\ & \mathsf{spk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ & \quad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ & \quad (c\_PrescriptID, c_{pt}\_Id_{ph}, vc_1, vc_2, vc_3, vc_3', vc_4, c_5)),\\ & \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}),\\ & c\_PrescriptID, c_{pt}\_Id_{ph}, vc_1, vc_2, vc_3, vc_3', vc_4, c_5) = \mathsf{true}.\end{aligned} \tag{R6}$$

From the modelling of each message exchanging in this sub-protocol, the behaviour of the patient and the pharmacist of this sub-protocol is obvious as $P_{pt}\_p_2$ in Figure 5 and $P_{ph}\_p_1$ Figure 6, respectively. Thus, this sub-protocol is modelled as $P_{pt}\_p_2 \mid P_{ph}\_p_1$.

Modelling Pharmacist-MPA sub-protocol The pharmacist-MPA sub-protocol is used for the pharmacist to report the received prescriptions to the MPA. There are two steps: first the pharmacist and the MPA authenticate each other, second, the pharmacist forwards prescription, encrypted patient information and the receipt to MPA. The communication of this sub-protocol is shown in Figure 6.2.

As the pharmacist mostly forwards the information supplied by the patient, this protocol greatly resembles the patient-pharmacist protocol described above. Each step in Figure 6.2 is modelled in details as follows:

1. The pharmacist authenticates himself to MPA by sending

$$\mathsf{sign}((\mathtt{Id}_{ph}, c_{ph}\_Id_{mpa}), \mathtt{sk}_{ph}).$$

2. MPA stores this authentication in $rcv_{mpa}\_Auth_{ph}$, which the MPA verifies against the pharmacist's public key,

$$\mathsf{Vfy\text{-}sign}(rcv_{mpa}\_Auth_{ph}, rcv_{mpa}\_pk_{ph}).$$

If the verification succeeds, according to the rule $R2$, the MPA then authenticates itself to the pharmacist by sending $\mathsf{sign}(\mathsf{Id}_{mpa}, \mathtt{sk}_{mpa})$.

3. The pharmacist verifies the MPA authentication rcv_Auth_{mpa},

$$\mathsf{Vfy\text{-}sign}(rcv\_Auth_{mpa}, \mathsf{pk}_{mpa}).$$

If the verification succeeds, according to the rule $R2$, the pharmacist sends the following to the MPA: prescription $rcv_{ph}\_PrescProof$, received receipt $rcv\_ReceiptAck$, and verifiable encryptions rcv_vc_1, rcv_vc_2, rcv_vc_3, rcv_vc_3', rcv_vc_4, rcv_c_5.

4. The MPA verifies the correctness of the information it received. The verification functions are the same as used in the process of pharmacist.

$$\begin{aligned}&(c_{mpa}\_Comt_{dr}, c_{mpa}\_Cred_{dr}) = \mathsf{getSpkVinfo}(rcv_{mpa}\_PrescProof)\\ &(c_{mpa}\_presc, c_{mpa}\_PrescriptID, = c_{mpa}\_Comt_{dr}, c_{mpa}\_Comt_{pt})\\ &\qquad\qquad = \mathsf{getSpkMsg}(rcv_{mpa}\_PrescProof)\\ &\mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(rcv_{mpa}\_PrescProof, (c_{mpa}\_Cred_{dr}, c_{mpa}\_presc, c_{mpa}\_PrescriptID,\\ &\qquad\qquad c_{mpa}\_Comt_{dr}, c_{mpa}\_Comt_{pt}))\end{aligned}$$

```
let $P_{pt}\_p_2 =$
            in(ch, $rcv\_Auth_{ph}$);
            if Vfy-sign($rcv\_Auth_{ph}$, $rcv_{pt}\_pk_{ph}$) = true then
            let (= $c_{pt}\_Id_{ph}$, $c_{pt}\_Id_{mpa}$) = getsignmsg($rcv\_Auth_{ph}$, $rcv_{pt}\_pk_{ph}$) in
            let $c_{pt}\_pk_{mpa}$ = key($c_{pt}\_Id_{mpa}$) in
            out(ch, zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc), Sss)));
            $\nu$nonce;
            let $vc_1$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                        (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                         enc(Hii, $c_{pt}\_pk_{mpa}$))) in
            let $vc_2$ = zk(($c\_Pnym_{dr}$, $rcv\_r_{dr}$), ($rcv\_PrescProof$,
                         enc($c\_Pnym_{dr}$, $c_{pt}\_pk_{mpa}$))) in
            let $vc_3$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                        (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                         enc(Pnym$_{pt}$, pk$_{sso}$))) in
            let $vc_3'$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                        (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                         enc(Hii, pk$_{sso}$))) in
            let $vc_4$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                        (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                         enc(Pnym$_{pt}$, $c_{pt}\_pk_{mpa}$))) in
            let $vc_5$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                        (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                         enc(Pnym$_{pt}$, $c_{pt}\_pk_{hii}$))) in
            let $c_5$ = enc($vc_5$, $c_{pt}\_pk_{mpa}$) in
            out(ch, ($rcv\_PrescProof$,
                  spk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc), commit(Id$_{pt}$, $r_{pt}$)),
                     nonce),
                  $vc_1$, $vc_2$, $vc_3$, $vc_3'$, $vc_4$, $c_5$));
            in(ch, $rcv\_Invoice$);
            let $ReceiptAck$ = spk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                                ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                                ($c\_PrescriptID$, $c_{pt}\_Id_{ph}$, $vc_1$, $vc_2$, $vc_3$, $vc_3'$, $vc_4$, $c_5$)) in
            out(ch, $ReceiptAck$).
```

Fig. 5. The patient process P_{pt} in Patient-Pharmacist sub-protocol.

```
let $P_{ph}\_p_1 =$
        $\mathsf{out}(\mathtt{ch}, \mathsf{sign}((\mathtt{Id}_{ph}, c_{ph}\_Id_{mpa}), \mathtt{sk}_{ph}));$
        $\mathsf{in}(\mathtt{ch}, rcv\_PtAuthSss);$
        let $(c_{ph}\_Cred_{pt}, c_{ph}\_Sss) = \mathsf{getpublic}(rcv\_PtAuthSss)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{PtAuthSss}}(rcv\_PtAuthSss, (c_{ph}\_Cred_{pt}, c_{ph}\_Sss)) = \mathsf{true}$ then
        $\mathsf{in}(\mathtt{ch}, (rcv_{ph}\_PrescProof, rcv_{ph}\_PtSpk,$
                $rcv\_vc_1, rcv\_vc_2, rcv\_vc_3, rcv\_vc'_3, rcv\_vc_4, rcv\_c_5));$
        let $(c_{ph}\_Comt_{dr}, c_{ph}\_Cred_{dr}) = \mathsf{getSpkVinfo}(rcv_{ph}\_PrescProof)$ in
        let $(c_{ph}\_presc, c_{ph}\_PrescriptID, = c_{ph}\_Comt_{dr}, c_{ph}\_Comt_{pt})$
            $= \mathsf{getSpkMsg}(rcv_{ph}\_PrescProof)$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(rcv_{ph}\_PrescProof, (c_{ph}\_Cred_{dr}, c_{ph}\_presc, c_{ph}\_PrescriptID,$
                                $c_{ph}\_Comt_{dr}, c_{ph}\_Comt_{pt})) = \mathsf{true}$ then
        let $c\_msg = \mathsf{getSpkMsg}(rcv_{ph}\_PtSpk)$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{PtSpk}}(rcv_{ph}\_PtSpk,$
                                $(c_{ph}\_Cred_{pt}, c_{ph}\_Comt_{pt}, c\_msg)) = \mathsf{true}$ then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_1) = \mathsf{getpublic}(rcv\_vc_1)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv\_vc_1, (c_{ph}\_Cred_{pt}, c\_Enc_1, rcv_{ph}\_pk_{mpa})) = \mathsf{true}$ then
        let $(= rcv_{ph}\_PrescProof, c\_Enc_2) = \mathsf{getpublic}(rcv\_vc_2)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(rcv\_vc_2, (rcv_{ph}\_PrescProof,$
                                        $c\_Enc_2, rcv_{ph}\_pk_{mpa})) = \mathsf{true}$ then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_3) = \mathsf{getpublic}(rcv\_vc_3)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv\_vc_3, (c_{ph}\_Cred_{pt}, c\_Enc_3, \mathtt{pk}_{sso})) = \mathsf{true}$ then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc'_3) = \mathsf{getpublic}(rcv\_vc'_3)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv\_vc'_3, (c_{ph}\_Cred_{pt}, c\_Enc'_3, \mathtt{pk}_{sso})) = \mathsf{true}$ then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_4) = \mathsf{getpublic}(rcv\_vc_4)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv\_vc_4,$
                                $(c_{ph}\_Cred_{pt}, c\_Enc_4, rcv_{ph}\_pk_{mpa})) = \mathsf{true}$ then
        $\mathsf{out}(\mathtt{ch}, \mathsf{invoice}(c_{ph}\_PrescriptID));$
        $\mathsf{in}(\mathtt{ch}, rcv\_ReceiptAck);$
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv\_ReceiptAck, (c_{ph}\_Cred_{pt}, c_{ph}\_PrescriptID,$
        $\mathtt{Id}_{ph}, rcv\_vc_1, rcv\_vc_2, rcv\_vc_3, rcv\_vc'_3, rcv\_vc_4, rcv\_c_5)) = \mathsf{true}$ then$0$.
```

Fig. 6. The pharmacist process P_{ph} in Patient-Pharmacist sub-protocol.

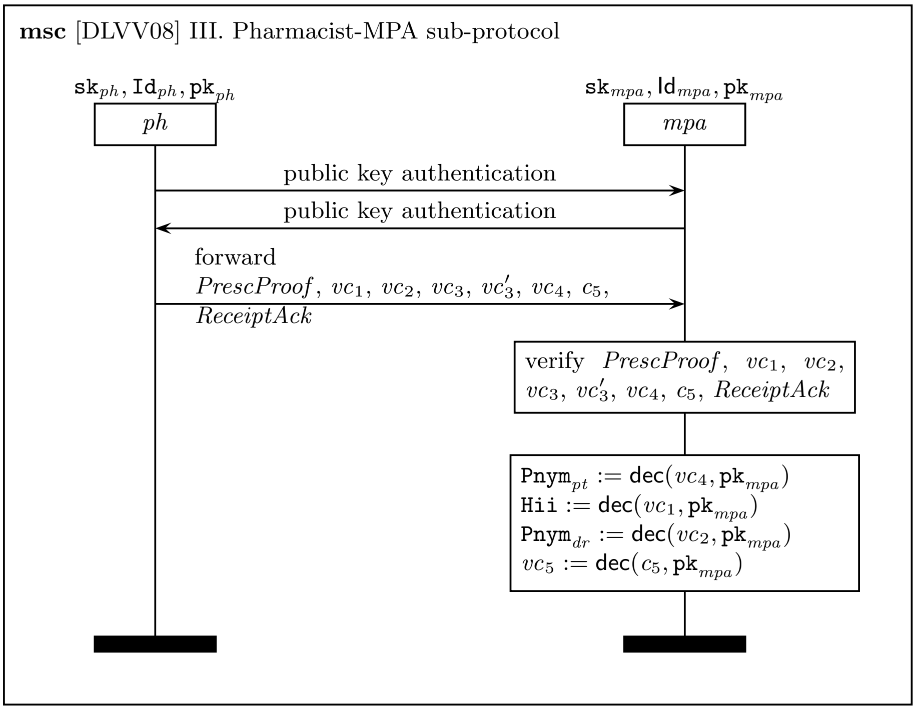

msc [DLVV08] III. Pharmacist-MPA sub-protocol
$\mathtt{sk}_{ph}, \mathtt{Id}_{ph}, \mathtt{pk}_{ph}$
ph
$\mathtt{sk}_{mpa}, \mathsf{Id}_{mpa}, \mathtt{pk}_{mpa}$
mpa
public key authentication
public key authentication
forward
$PrescProof$, vc_1, vc_2, vc_3, vc'_3, vc_4, c_5, $ReceiptAck$
verify $PrescProof$, vc_1, vc_2, vc_3, vc'_3, vc_4, c_5, $ReceiptAck$
$\mathtt{Pnym}_{pt} := \mathsf{dec}(vc_4, \mathtt{pk}_{mpa})$
$\mathtt{Hii} := \mathsf{dec}(vc_1, \mathtt{pk}_{mpa})$
$\mathtt{Pnym}_{dr} := \mathsf{dec}(vc_2, \mathtt{pk}_{mpa})$
$vc_5 := \mathsf{dec}(c_5, \mathtt{pk}_{mpa})$

Fig. 7. Pharmacist-MPA sub-protocol

$$
\begin{array}{l}
(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_1) = \mathsf{getpublic}(rcv_{mpa}\_vc_1) \\
(= rcv_{mpa}\_PrescProof, c_{mpa}\_Enc_2) = \mathsf{getpublic}(rcv_{mpa}\_vc_2) \\
(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc3) = \mathsf{getpublic}(rcv_{mpa}\_vc_3) \\
(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc'_3) = \mathsf{getpublic}(rcv_{mpa}\_vc'_3) \\
(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_4) = \mathsf{getpublic}(rcv_{mpa}\_vc_4)
\end{array}
$$

$$
\begin{array}{l}
\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv_{mpa}\_vc_1, (c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_1, \mathtt{pk}_{mpa})) \\
\mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(rcv_{mpa}\_vc_2, (rcv_{mpa}\_PrescProof, c_{mpa}\_Enc_2, \mathtt{pk}_{mpa})) \\
\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv_{mpa}\_vc_3, (c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_3, \mathtt{pk}_{sso})) \\
\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv_{mpa}\_vc'_3, (c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc'_3, \mathtt{pk}_{sso})) \\
\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv_{mpa}\_vc_4, (c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_4, \mathtt{pk}_{mpa}))
\end{array}
$$

$$
\begin{array}{l}
\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv_{mpa}\_ReceiptAck, (c_{mpa}\_Cred_{pt}, c_{mpa}\_PrescriptID, \\
\quad c_{mpa}\_Id_{ph}, rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2, rcv_{mpa}\_vc_3, rcv_{mpa}\_vc_4, rcv_{mpa}\_c_5))
\end{array}
$$

If the verifications succeed, according to rules $R1$, $R3$, $R4$, $R5$, $R3$ and $R5$, respectively, the MPA decrypts $rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2, rcv_{mpa}\_vc_4$ and $rcv_{mpa}\_c_5$, obtains the patient's HII, the doctor pseudonym, the patient pseudonym and $rcv_{mpa}\_vc_5$ as follows:

$$
\begin{aligned}
c_{mpa}\_Hii &= \mathsf{dec}(rcv_{mpa}\_vc_1, \mathtt{sk}_{mpa}) \\
c_{mpa}\_Pnym_{dr} &= \mathsf{dec}(rcv_{mpa}\_vc_2, \mathtt{sk}_{mpa}) \\
c_{mpa}\_Pnym_{pt} &= \mathsf{dec}(rcv_{mpa}\_vc_4, \mathtt{sk}_{mpa}) \\
rcv_{mpa}\_vc_5 &= \mathsf{dec}(rcv_{mpa}\_c_5, \mathtt{sk}_{mpa}).
\end{aligned}
$$

The storing information to database is beyond our concern.

```
let $P_{ph}\_p_2$ =
            out(ch, (sign(($\mathtt{Id}_{ph}$, $c_{ph}\_Id_{mpa}$), $\mathtt{sk}_{ph}$), $\mathtt{Id}_{ph}$));
            in(ch, $rcv\_Auth_{mpa}$);
            if Vfy-sign($rcv\_Auth_{mpa}$, $rcv_{ph}\_pk_{mpa}$) = true then
            out(ch, ($rcv_{ph}\_PrescProof$,
                  $rcv\_vc_1$, $rcv\_vc_2$, $rcv\_vc_3$, $rcv\_vc'_3$, $rcv\_vc_4$, $rcv\_c_5$,
                  $rcv\_ReceiptAck$))
```

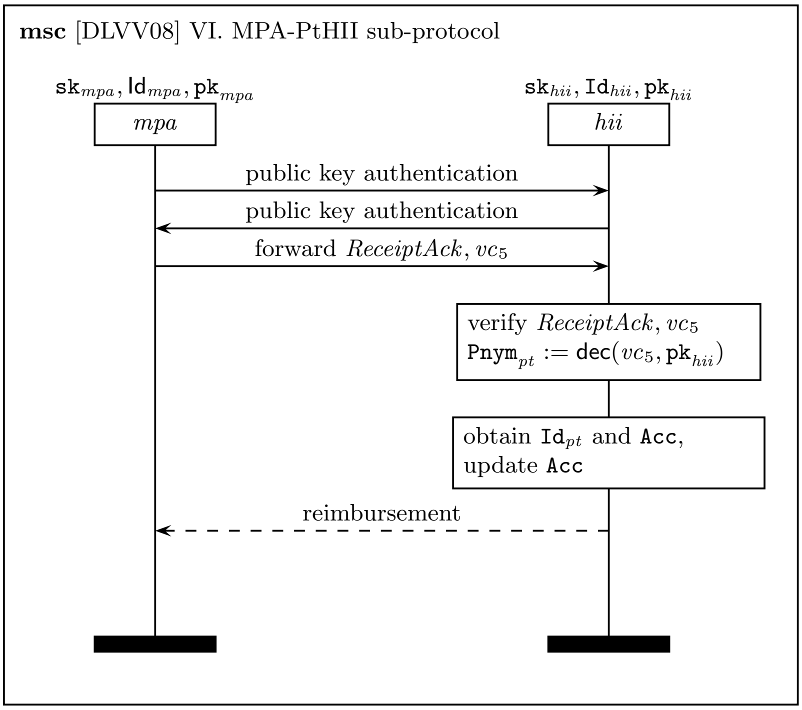
Fig. 8. The pharmacist process P_{ph} in Pharmacist-MPA sub-protocol.

Modelling MPA-HII sub-protocol. This protocol covers the exchange of information between the pharmacist's MPA and the patient's HII. The goal of this protocol is for the MPA to be reimbursed by the patient's HII. The communication of this sub-protocol consists of two steps: first, the MPA and

```
let $P_{mpa}\_p_1 =$
        $\mathsf{in}(\mathtt{ch}, (rcv_{mpa}\_Auth_{ph}, c_{mpa}\_Id_{ph}));$
        let $rcv_{mpa}\_pk_{ph} = \mathsf{key}(c_{mpa}\_Id_{ph})$ in
        $\mathsf{Vfy\text{-}sign}(rcv_{mpa}\_Auth_{ph}, rcv_{mpa}\_pk_{ph}) = \mathsf{true}$
        let $(= c_{mpa}\_Id_{ph}, = \mathsf{Id}_{mpa})$
            $= \mathsf{getSpkMsg}(rcv_{mpa}\_Auth_{ph}, rcv_{mpa}\_pk_{ph})$ in
        $\mathsf{out}(\mathtt{ch}, \mathsf{sign}(\mathsf{Id}_{mpa}, \mathsf{sk}_{mpa}));$
        $\mathsf{in}(\mathtt{ch}, (rcv_{mpa}\_PrescProof, rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2, rcv_{mpa}\_vc_3,$
                $rcv_{mpa}\_vc'_3, rcv_{mpa}\_vc_4, rcv_{mpa}\_c_5, rcv_{mpa}\_ReceiptAck));$
        let $(c_{mpa}\_Comt_{dr}, c_{mpa}\_Cred_{dr}) = \mathsf{getSpkVinfo}(rcv_{mpa}\_PrescProof)$ in
        let $(c_{mpa}\_presc, c_{mpa}\_PrescriptID, = c_{mpa}\_Comt_{dr}, c_{mpa}\_Comt_{pt})$
            $= \mathsf{getSpkMsg}(rcv_{mpa}\_PrescProof)$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(rcv_{mpa}\_PrescProof, (c_{mpa}\_Cred_{dr}, c_{mpa}\_presc,$
                    $c_{mpa}\_PrescriptID, c_{mpa}\_Comt_{dr}, c_{mpa}\_Comt_{pt})) = \mathsf{true}$ then
        let $(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_1) = \mathsf{getpublic}(rcv_{mpa}\_vc_1)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv_{mpa}\_vc_1,$
                        $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_1, \mathsf{pk}_{mpa})) = \mathsf{true}$ then
        let $c_{mpa}\_Hii = \mathsf{dec}(c_{mpa}\_Enc_1, \mathsf{sk}_{mpa})$ in
        let $(= rcv_{mpa}\_PrescProof, c_{mpa}\_Enc_2) = \mathsf{getpublic}(rcv_{mpa}\_vc_2)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(rcv_{mpa}\_vc_2,$
                            $(rcv_{mpa}\_PrescProof, c_{mpa}\_Enc_2, \mathsf{pk}_{mpa})) = \mathsf{true}$ then
        let $c_{mpa}\_Pnym_{dr} = \mathsf{dec}(c_{mpa}\_Enc_2, \mathsf{sk}_{mpa})$ in
        let $(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc3) = \mathsf{getpublic}(rcv_{mpa}\_vc_3)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv_{mpa}\_vc_3,$
                            $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_3, \mathsf{pk}_{sso})) = \mathsf{true}$ then
        let $(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc'_3) = \mathsf{getpublic}(rcv_{mpa}\_vc'_3)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv_{mpa}\_vc'_3,$
                        $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc'_3, \mathsf{pk}_{sso})) = \mathsf{true}$ then
        let $(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_4) = \mathsf{getpublic}(rcv_{mpa}\_vc_4)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv_{mpa}\_vc_4,$
                            $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_4, \mathsf{pk}_{mpa})) = \mathsf{true}$ then
        let $c_{mpa}\_Pnym_{pt} = \mathsf{dec}(c_{mpa}\_Enc_4, \mathsf{sk}_{mpa})$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv_{mpa}\_ReceiptAck, (c_{mpa}\_Cred_{pt},$
            $c_{mpa}\_PrescriptID, c_{mpa}\_Id_{ph}, rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2, rcv_{mpa}\_vc_3,$
            $rcv_{mpa}\_vc'_3, rcv_{mpa}\_vc_4, rcv_{mpa}\_c_5))$
          $= \mathsf{true}$ then $0$.
```

Fig. 9. The MPA process P_{mpa} in Pharmacist-MPA sub-protocol.

the HII authenticate each other, second, the MPA forwards the receipt and the encrypted patient pseudonym to the HII. Note that reimbursement is not in the scope of the protocol. The communication is as shown in Figure 6.2.



msc [DLVV08] VI. MPA-PtHII sub-protocol
$\mathtt{sk}_{mpa}, \mathsf{Id}_{mpa}, \mathtt{pk}_{mpa}$
$\mathtt{sk}_{hii}, \mathtt{Id}_{hii}, \mathtt{pk}_{hii}$
mpa
hii
public key authentication
public key authentication
forward $ReceiptAck, vc_5$
verify $ReceiptAck, vc_5$
$\mathtt{Pnym}_{pt} := \mathsf{dec}(vc_5, \mathtt{pk}_{hii})$
obtain $\mathtt{Id}_{pt}$ and $\mathtt{Acc}$,
update $\mathtt{Acc}$
reimbursement

Fig. 10. MPA-PtHII sub-protocol

As shown in Figure 6.2, the detailed modelling of each step is as follows:

1. The MPA authenticates to the HII using public key authentication,

$$\mathsf{sign}(\mathsf{Id}_{mpa}, \mathtt{sk}_{mpa}).$$

2. The HII stores the authentication in $rcv_{hii}\_Auth_{mpa}$ and verifies it as follows:

$$\mathsf{Vfy\text{-}sign}(rcv_{hii}\_Auth_{mpa}, \mathtt{pk}_{mpa}).$$

If the verification succeeds according to rule $R2$, the HII authenticates to the MPA using public key authentication,

$$\mathsf{sign}(\mathtt{Id}_{hii}, \mathtt{sk}_{hii}).$$

3. The MPA stores the authentication in $rcv_{mpa}\_Auth_{hii}$ and verifies it as follows:

$$\mathsf{Vfy\text{-}sign}(rcv_{mpa}\_Auth_{hii}, \mathtt{pk}_{hii}).$$

If the verification succeeds according to rule $R2$, the MPA sends the receipt $rcv_{mpa}\_PrescProof$ and the patient pseudonym encrypted for the HII $rcv_{mpa}\_vc_5$.

4. The HII receives them as $rcv_{hii}\_ReceiptAck$ and $c_{hii}\_vc_5$, and verifies their correctness as follows:

$$\begin{aligned}
&c_{hii}\_Cred_{pt} = \mathsf{getSpkVinfo}(rcv_{hii}\_ReceiptAck)\\
&(c_{hii}\_PrescriptID, c_{hii}\_Id_{ph}, c_{hii}\_vc_1, c_{hii}\_vc_2, c_{hii}\_vc_3, c_{hii}\_vc_3', c_{hii}\_vc_4, c_{hii}\_c_5)\\
&\qquad\qquad\qquad\qquad\qquad\qquad = \mathsf{getSpkMsg}(rcv_{hii}\_ReceiptAck)\\
&\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv_{hii}\_ReceiptAck, (c_{hii}\_Cred_{pt}, c_{hii}\_PrescriptID, c_{hii}\_Id_{ph},\\
&\qquad\qquad\qquad c_{hii}\_vc_1, c_{hii}\_vc_2, c_{hii}\_vc_3, c_{hii}\_vc_3', c_{hii}\_vc_4, c_{hii}\_c_5))
\end{aligned}$$

$$\begin{aligned}
&(= c_{hii}\_Cred_{pt}, c_{hii}\_Enc_5) = \mathsf{getpublic}(c_{hii}\_vc_5)\\
&\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(c_{hii}\_vc_5, (c_{hii}\_Cred_{pt}, c_{hii}\_Enc_5, \mathtt{pk}_{hii}))
\end{aligned}$$

If the verifications succeed, according to rules $R6$ and $R5$, respectively, the HII decrypts $c_{hii}\_vc_5$ and obtains the patient's pseudonym.

$$c_{hii}\_Pnym_{pt} = \mathsf{dec}(c_{hii}\_Enc_5, \mathtt{sk}_{hii}).$$

Afterwards, the HII pays the MPA and updates the patient account. As before, handling payment and storing information are beyond the scope of the DLVV08 protocol and therefore, we do not model this stage.

Following the modelling of each step, the behaviour of MPA and HII in this sub-protocol is modelled as $P_{mpa}\_p_2$ in Figure 11 and P_{hii} in Figure 12, respectively.

```
let $P_{mpa}\_p_2$ =
            $\mathsf{out}(\mathtt{ch}, (\mathsf{sign}(\mathsf{Id}_{mpa}, \mathtt{sk}_{mpa}), \mathsf{Id}_{mpa}))$;
            $\mathsf{in}(\mathtt{ch}, rcv_{mpa}\_Auth_{hii})$;
            let $c_{mpa}\_pk_{hii} = \mathsf{key}(c_{mpa}\_Hii)$ in
            if $\mathsf{Vfy\text{-}sign}(rcv_{mpa}\_Auth_{hii}, c_{mpa}\_pk_{hii})$ = true then
            if $\mathsf{getsignmsg}(rcv_{mpa}\_Auth_{hii}, c_{mpa}\_pk_{hii}) = c_{mpa}\_Hii$ then
            $\mathsf{out}(\mathtt{ch}, (rcv_{mpa}\_ReceiptAck, \mathsf{dec}(rcv_{mpa}\_c_5, \mathtt{sk}_{mpa})))$;
            $\mathsf{in}(\mathtt{ch}, rcv_{mpa}\_Invoice)$.
```

Fig. 11. The MPA process P_{mpa} in MPA-HII sub-protocol.

The protocol In summary, the DLVV08 protocol is modelled as five roles R_{dr}, R_{pt}, R_{ph}, R_{mpa}, and R_{hii} running in parallel.

$$DLV = \nu\tilde{m}.init.(!R_{pt} \;|!R_{dr} \;|!R_{ph} \;|!R_{mpa} \;|!R_{hii})$$

where $\nu\tilde{m}$ represents global secrets $\mathtt{sk}_{sso}$ and private channels $\mathtt{ch}_{hp}$, $\mathtt{ch}_{hm}$, $\mathtt{ch}_{pm}$, $\mathtt{ch}_{mp}$, $\mathtt{ch}_{phpt}$, $\mathtt{ch}_{mh}$; process *init* initialise the settings of the protocol and the

```
let $P_{hii}$ =
        $\mathsf{in}(\mathtt{ch}, (rcv_{hii}\_Auth_{mpa}, rcv_{hii}\_Id_{mpa}))$;
        let $c_{hii}\_pk_{mpa} = \mathsf{key}(rcv_{hii}\_Id_{mpa})$ in
        if $\mathsf{Vfy\text{-}sign}(rcv_{hii}\_Auth_{mpa}, c_{hii}\_pk_{mpa}) = \mathtt{true}$ then
        $\mathsf{out}(\mathtt{ch}, \mathsf{sign}(\mathtt{Id}_{hii}, \mathtt{sk}_{hii}))$;
        $\mathsf{in}(\mathtt{ch}, (rcv_{hii}\_ReceiptAck, c_{hii}\_vc_5))$;
        let $c_{hii}\_Cred_{pt} = \mathsf{getSpkVinfo}(rcv_{hii}\_ReceiptAck)$ in
        let $(c_{hii}\_PrescriptID, c_{hii}\_Id_{ph}, c_{hii}\_vc_1, c_{hii}\_vc_2, c_{hii}\_vc_3, c_{hii}\_vc_3', c_{hii}\_vc_4,$
                $c_{hii}\_c_5) = \mathsf{getSpkMsg}(rcv_{hii}\_ReceiptAck)$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv_{hii}\_ReceiptAck, (c_{hii}\_Cred_{pt},$
            $c_{hii}\_PrescriptID, c_{hii}\_Id_{ph}, c_{hii}\_vc_1, c_{hii}\_vc_2, c_{hii}\_vc_3, c_{hii}\_vc_3',$
            $c_{hii}\_vc_4, c_{hii}\_c_5)) = \mathtt{true}$ then
        let $(= c_{hii}\_Cred_{pt}, c_{hii}\_Enc_5) = \mathsf{getpublic}(c_{hii}\_vc_5)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(c_{hii}\_vc_5, (c_{hii}\_Cred_{pt}, c_{hii}\_Enc_5, \mathtt{pk}_{hii})) = \mathsf{true}$
        then
        let $c_{hii}\_Pnym_{pt} = \mathsf{dec}(c_{hii}\_Enc_5, \mathtt{sk}_{hii})$ in
        $\mathsf{out}(\mathtt{ch}, \mathsf{invoice}(c_{hii}\_PrescriptID))$.
```

Fig. 12. The HII process P_{hii}.

communication partner of each user.

$$init := \text{let } \mathtt{pk}_{sso} = \mathsf{pk}(\mathtt{sk}_{sso}) \text{ in } \mathsf{out}(\mathtt{ch}, \mathtt{pk}_{sso});$$

Each role is modelled as $R_i{:=}init_i; !P_i$. We notice that the patient process, pharmacist process and MPA process has two parts. Since the DLVV08 protocol works as four sub-protocols executing in order, as shown in Figure 6.2, Thus, we compose $P_{pt}\_p_1$ and $P_{pt}\_p_2$ to obtain P_{pt}, compose $P_{ph}\_p_1$ and $P_{ph}\_p_2$ to obatin P_{ph}, and compose $P_{mpa}\_p_1$ and $P_{mpa}\_p_2$ to obtain P_{mpa}. In details, each role is modelled as follows:

7 Analysis

We first analyse security and privacy properties of patients and doctors including standard secrecy, authentication, anonymity and untraceablity. We find that with the assumptions in Section 5.4, the DLVV08 protocol does not satisfy secrecy of a patient's social security status and a doctor's pseudonym; does not satisfies patient authentication; does not satisfy doctor anonymity; and does not satisfy patient and doctor untraceability. Then we analyse the enforced privacy properties and find that the protocol does not satisfy enforced prescribing-privacy and pharmacist independency of enforced prescribing-privacy.

7.1 Analysis of security and privacy properties

Patient and doctor secrecy The DLVV08 protocol is claimed to satisfy the requirement: any party involved in the prescription processing work flow should

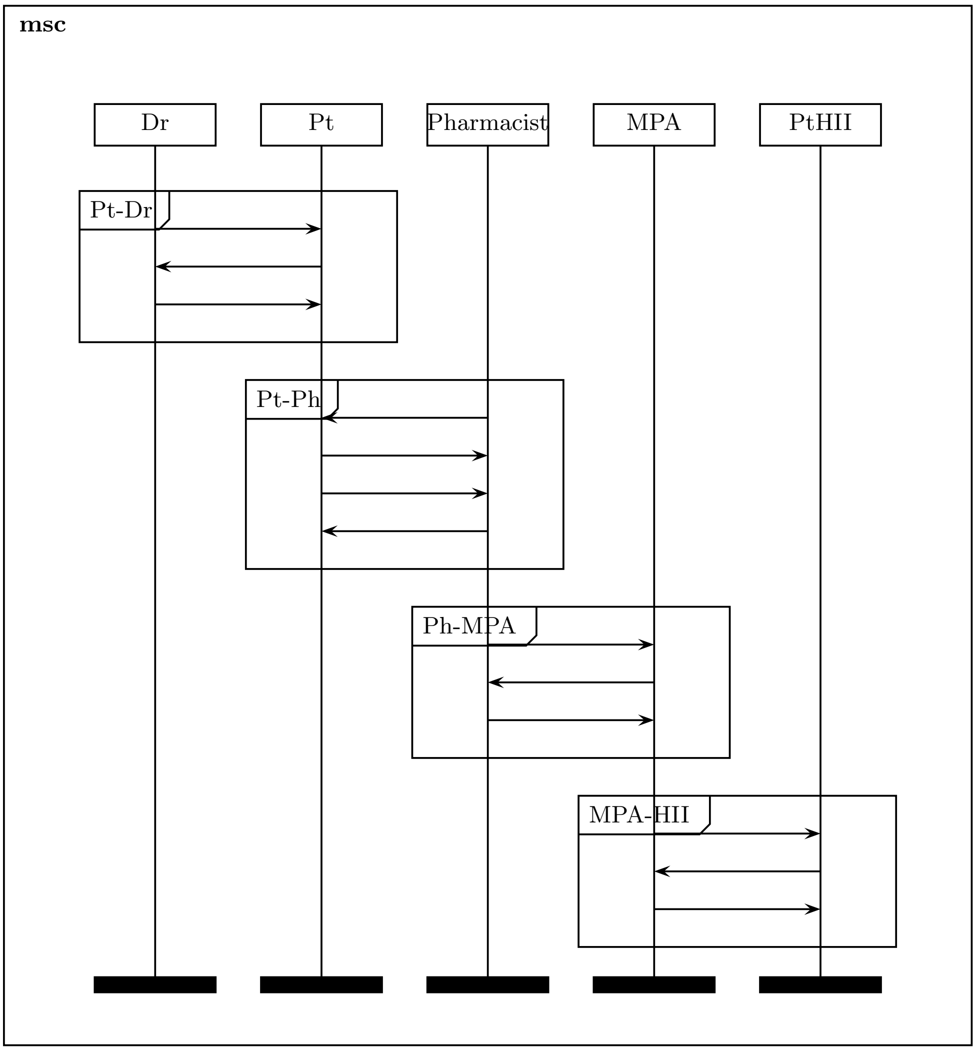


Fig. 13. The overview of DLVV08 protocol

$$\begin{array}{ll} \left.\begin{array}{l} R_{dr} := \nu\mathtt{Id}_{dr}; \\ \qquad\quad \nu\mathtt{Pnym}_{dr}; \end{array}\right\} & init_{dr} \\ \qquad\quad !(P_{dr}) & \end{array}$$

Fig. 14. Process R_{dr}.

$$
\begin{array}{l}
R_{pt} := \nu\mathtt{Id}_{pt};\\
\qquad \left.\begin{array}{l} \nu\mathtt{Pnym}_{pt};\nu\mathtt{Sss};\nu\mathtt{Acc};\\ \mathsf{in}(\mathsf{ch}_{hp},\mathtt{Hii});\mathtt{let}\ \ c_{pt}\_pk_{hii}=\mathsf{key}(\mathtt{Hii})\ \ \mathtt{in}\end{array}\right\}init_{pt}\\
\qquad \left.\begin{array}{l} !(\mathsf{in}(\mathsf{ch}_{phpt}, rcv\_pk_{ph});\\ \ \mathtt{let}\ \ rcv_{pt}\_pk_{ph} = rcv\_pk_{ph}\ \ \mathtt{in}\ \mathtt{let}\ \mathtt{Id}_{ph}=\mathsf{host}(rcv\_pk_{ph})\ \ \mathtt{in}\\ \left.\begin{array}{l}(\ldots\\ \mathtt{let}\ \ c\_Pnym_{dr}=\mathsf{open}(c\_Comt_{dr}, rcv\_r_{dr})\ \ \mathtt{in}\end{array}\right\}P_{pt}\_p_1\\ \left.\begin{array}{l}\mathsf{in}(\mathsf{ch}, rcv\_Auth_{ph});\\ \ldots)\end{array}\right\}P_{pt}\_p_2 \end{array}\right\}P_{pt}
\end{array}
$$

Fig. 15. Process R_{pt}.

$$
\begin{array}{l}
R_{ph} := \nu\mathsf{sk}_{ph};\\
\qquad \mathtt{let}\ \ \mathsf{pk}_{ph} = \mathsf{pk}(\mathsf{nsk}_{ph})\ \ \mathtt{in}\\
\qquad \mathtt{let}\ \ \mathtt{Id}_{ph} = \mathsf{host}(\mathsf{pk}_{ph})\ \ \mathtt{in}\\
\qquad (!\mathsf{out}(\mathsf{ch}, \mathsf{pk}_{ph})\ |!(\mathsf{out}(\mathsf{ch}_{phpt}, \mathsf{pk}_{ph}))\ |\\
\qquad \left.\begin{array}{l} \ !(\mathsf{in}(\mathsf{ch}_{mp}, rcv_{ph}\_pk_{mpa});\\ \qquad \mathtt{let}\ \ c_{ph}\_Id_{mpa} = \mathsf{host}(rcv_{ph}\_pk_{mpa})\ \ \mathtt{in}\\ \left.\begin{array}{l} (\ldots\\ \mathtt{if}\ \ \mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv\_ReceiptAck, (c_{ph}\_Cred_{pt},\\ \qquad c_{ph}\_PrescriptID, \mathtt{Id}_{ph}, rcv\_vc_1, rcv\_vc_2, rcv\_vc_3,\\ \qquad rcv\_vc'_3, rcv\_vc_4, rcv\_c_5)) = \mathsf{true}\ \mathtt{then}\end{array}\right\}P_{ph}\_p_1\\ \left.\begin{array}{l} \mathsf{out}(\mathsf{ch}, (\mathsf{sign}((\mathtt{Id}_{ph}, c_{ph}\_Id_{mpa}), \mathsf{sk}_{ph}), \mathtt{Id}_{ph}));\\ \ldots)\end{array}\right\}P_{ph}\_p_2 \end{array}\right\}P_{ph}
\end{array}
$$

Fig. 16. Process R_{ph}.

$$
\begin{array}{l}
R_{mpa} := \nu\mathsf{sk}_{mpa};\\
\qquad \mathtt{let}\ \ \mathsf{pk}_{mpa} = \mathsf{pk}(\mathsf{sk}_{mpa})\ \ \mathtt{in}\\
\qquad \mathtt{let}\ \ \mathsf{Id}_{mpa} = \mathsf{host}(\mathsf{pk}_{mpa})\ \ \mathtt{in}\\
\qquad (!\mathsf{out}(\mathsf{ch}, \mathsf{pk}_{mpa})\ |!\mathsf{out}(\mathsf{ch}_{mp}, \mathsf{pk}_{mpa})\ |\\
\qquad \left.\begin{array}{l} \left.\begin{array}{l} \ !(\ldots\\ \ \mathtt{if}\ \ \mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv_{mpa}\_ReceiptAck, (c_{mpa}\_Cred_{pt},\\ \qquad c_{mpa}\_PrescriptID, c_{mpa}\_Id_{ph}, rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2,\\ \qquad rcv_{mpa}\_vc_3, rcv_{mpa}\_vc'_3, rcv_{mpa}\_vc_4, rcv_{mpa}\_c_5))\\ \qquad = \mathsf{true}\ \mathtt{then}\end{array}\right\}P_{mpa}\_p_1\\ \left.\begin{array}{l} \ \mathsf{out}(\mathsf{ch}, (\mathsf{sign}(\mathsf{Id}_{mpa}, \mathsf{sk}_{mpa}), \mathsf{Id}_{mpa}));\\ \ \ldots)\end{array}\right\}P_{mpa}\_p_2 \end{array}\right\}P_{mpa}
\end{array}
$$

Fig. 17. Process R_{mpa}.

$$
\begin{array}{l}
R_{hii} := \nu\mathsf{sk}_{hii};\\
\qquad \mathtt{let}\ \ \mathsf{pk}_{hii} = \mathsf{pk}(\mathsf{sk}_{hii})\ \ \mathtt{in}\\
\qquad \mathtt{let}\ \ \mathtt{Id}_{hii} = \mathsf{host}(\mathsf{pk}_{hii})\ \ \mathtt{in}\\
\qquad (!\mathsf{out}(\mathsf{ch}, \mathsf{pk}_{hii})\ |!\mathsf{out}(\mathsf{ch}_{hp}, \mathtt{Id}_{hii})\ |!(P_{hii}))
\end{array}
$$

Fig. 18. Process R_{hii}.

$$\begin{array}{l} \mathit{DLV} = \\ \qquad \nu \mathtt{sk}_{sso}; \nu \mathtt{ch}_{hp}; \nu \mathtt{ch}_{mp}; \nu \mathtt{ch}_{phpt}; \\ \qquad \mathtt{let}\ \mathtt{pk}_{sso} = \mathtt{pk}(\mathtt{sk}_{sso})\ \mathtt{in} \\ \qquad \mathtt{out}(\mathtt{ch}, \mathtt{pk}_{sso}); \\ \qquad (!(R_{dr})\ |!(R_{pt})\ |!(R_{ph})\ |!(R_{mpa})\ |!(R_{hii})) \end{array}$$

Fig. 19. The DLVV08 protocol.

not know the information of a patient and a doctor unless the information is intended to be revealed in the protocol. In [10], this requirement is considered as an access control requirement. Moreover, it seems that the involved parties are assumed to be honest, although it is not clear whether the assumption is made. We argue that the requirement may not be satisfied when a involved party is dishonest. A user who should not access to a piece of information, may obtain the information by observing the Internet and manipulating the protocol. Thus, we consider secrecy of patient and doctor information respecting to the the Dolev-Yao adversary, who controls the whole Internet.

Private information of patients and doctors, which is considered for this requirement in the protocol, are as follows: patient identity ($\mathtt{Id}_{pt}$), doctor identity ($\mathtt{Id}_{dr}$), patient pseudonym ($\mathtt{Pnym}_{pt}$), doctor pseudonym ($\mathtt{Pnym}_{dr}$), patient social security status ($\mathtt{Sss}$), patient's health insurance institute ($\mathtt{Hii}$). We notice that, besides of the above information, a patient also has another piece of information: health expense account $\mathtt{Acc}$. We verify it as well. The requirement is modelled as standard secrecy of these information. When a protocol satisfies standard secrecy of a piece of information, the information is secret respecting to the Dolev-Yao adversary.

To verify secrecy of information, we need to model honest parties in the protocol. Obviously, if a protocol intends to reveal a piece of information to a party and the party is dishonest (revealing the information), the information is not secret. Thus, those parties are considered to be honest in the model. Since the prescription precessing goes through the whole protocol, the requirement should be satisfied over the whole protocol. We model one honest patient, one honest doctor, one honest pharmacist, one honest MPA and one honest HII in the protocol. The patient has private information $\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, $\mathtt{Sss}$, $\mathtt{Hii}$, and $\mathtt{Acc}$. The doctor has private information $\mathtt{Id}_{dr}$ and $\mathtt{Pnym}_{dr}$. We use 'attack: M' to query standard secrecy of M in ProVerif [12].

Verification result. A patient's identity, pseudonym, health expense account, health insurance institute and identity of a doctor ($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, $\mathtt{Hii}$ $\mathtt{Acc}$, $\mathtt{Id}_{dr}$) satisfy standard secrecy. A patient's social security status $\mathtt{Sss}$ and a doctor's pseudonym $\mathtt{Pnym}_{dr}$ do not satisfy standard secrecy. The $\mathtt{Sss}$ is revealed during the communication between the patient and the pharmacist. Although the protocol intends to reveal $\mathtt{Sss}$ to the pharmacist, other parties such as the doctor and the HII should not know it according to the access control of the protocol. However, if the doctor observes the Internet, he can access to the patient's social security status. The $\mathtt{Pnym}_{dr}$ is revealed during the communication between the patient and the doctor. According to the access control matrix in [10], a patient should not access to the doctor's pseudonym. However, in the protocol, the doctor first makes a commitment of his pseudonym and later sends to the patient the open information to the commitment. Thus, the patient knows the doctor's pseudonym. Other parties who are not supposed to know the doctor's pseudonym can access to it as well, if these parties observe the Internet.

Patient and doctor authentication The protocol claims that all parties should be able to properly authenticate each other. Comparing to authentications of pharmacists, MPA and HII, patient and doctor authentication are our focus, because patients and doctors use anonymous authentication. In this part, only patient and doctor authentications during the patient-doctor sub-protocol are explained in details. Authentications between patients and pharmacists are sketched.

The authenticate of a patient, means that when a doctor finishes his process and believes that he prescribed medicines for a patient, then the patient did ask the doctor for prescription. Similarly, the authentication of a doctor means that when a patient believes he visited a doctor, the doctor did prescribe medicines for the patient. Authentications are modelled as correspondence properties in ProVerif [24]. Non-injective correspondence '$\text{ev:}\mathit{EndEvent} \Longrightarrow \text{ev:}\mathit{StartEvent}$' means that if the *EndEvent* is executed, there must be an event *StartEvent* executed before. Injective correspondence '$\text{evinj:}\mathit{EndEvent} \Longrightarrow \text{evinj:}\mathit{StartEvent}$' means that if the event *EndEvent* is executed, there must be exactly one event *StartEvent* executed before. The difference between non-injective correspondence and injective correspondence is that, non-injective correspondence allows message replaying, while injective correspondence does not. It is not clear in [10] which type of authentication is required. We use injective correspondence property to model doctor and patient authentication, with the following thinking in mind: when a patient obtains a prescription from a doctor, the prescription should be generated only for this execution; when a doctor prescribes medicines for a patient, the patient should be a real one instead of the adversary replying a patient's information.

To verify the authentication of a patient, we add an event

$$\mathsf{EndDr}(c\_Cred_{pt}, c\_Comt_{pt})$$

at the end of the doctor process, meaning that the doctor believes that he prescribed medicines for a patient who has a credential c_Cred_{pt} and make a commitment c_Comt_{pt}, and add an event

$$\mathsf{StartPt}(\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}))$$

in the patient process, meaning that the patient process, in which the patient uses
$\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})$, and $\mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})$, is executed, i.e. the

patient did ask for prescription. Then we query

$$\textit{evinj} : \mathsf{EndDr}(x, y) ==> \textit{evinj} : \mathsf{StartPt}(x, y),$$

meaning that when the event EndDr is executed, there is an unique event StartPt has been executed before. This model means that when the doctor believes that he prescribed medicines for a patient identified by c_Cred_{pt} and c_Comt_{pt}, the patient who created c_Cred_{pt} and c_Comt_{pt} did ask for a prescription.

To authenticate a doctor, we add event

$$\mathsf{EndPt}(c\_Cred_{dr}, c\_Comt_{dr}, c\_presc, c\_PrescriptID)$$

at the end of the patient process, add event

$$\mathsf{StartDr}(\mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathtt{presc}, \textit{PrescriptID})$$

at the doctor process, and query

$$\textit{evinj} : \mathsf{EndPt}(x, y, z, t) ==> \textit{evinj} : \mathsf{StartDr}(x, y, z, t).$$

This means that when a patient believes that he obtains a prescription (indicated by c_presc and $c\_PrescriptID$) from a patient c_Comt_{dr}, the doctor did prescribe the medicines and made the commitment.

Note that an authentication here not only authenticates a patient or a doctor but also authenticates the communication messages belonging to the patient or the doctor. This is guaranteed by putting the communication messages as the parameters of events.

Verification result. The doctor authentication (injective and non-injective) succeeds. The non-injective patient authentication succeeds and injective patient authentication fails. The injective patient authentication fails, because when the adversary observes messages from a patient, the adversary can block the messages and replay old patient messages to impersonate a patient. The failure of patient authentication allows the adversary to replace a patient's authentication message with a fake one, thus the doctor prescribes medicines for a wrong patient. At the end of the the patient-doctor communication, the protocol does not clearly say that the patient checks the prescription information. If the patient does not check whether the patient commitment in the prescription is the same as his commitment, the patient cannot obtain medicines later from the pharmacist because of the patient commitment error in the protocol. Even we assume that the patient checks the prescription information, thus can find errors in the prescription immediately, the patient and the doctor have to run the protocol again.

We also verified the authentications of patients and pharmacists during the communication between a patient and a pharmacist. The patient authentication satisfies non-injective bu not injective correspondence property. This means that the messages received by a pharmacist are from a patient, but not necessarily from the current communication with the patient. The pharmacist authentication does not satisfy non-injective nor injective correspondence property. This means that the adversary can record messages from a pharmacist, pretend to be that pharmacist.

checked Security&Auth property	**initial model**	**cause(s)**	**impro**
Secrecy of $\mathtt{Id}_{pt}$	√		
Secrecy of $\mathtt{Pnym}_{pt}$	√		
Secrecy of $\mathtt{Sss}$	×	revealed	sessio
Secrecy of $\mathtt{Hii}$	√		
Secrecy of $\mathtt{Acc}$	√		
Secrecy of $\mathtt{Id}_{dr}$	√		
Secrecy of $\mathtt{Pnym}_{dr}$	×	revealed	sessio
Auth doctor to patient (inject)	√		
Auth doctor to patient (non-inject)	√		
Auth patient to doctor (inject)	×	replay attack	add ch
Auth patient to doctor (non-inject)	√		
Auth pharmacist to patient (inject)	×	adv. can reply 1st message, compute 2nd message	sign the
Auth pharmacist to patient (non-inject)	×	adv. can reply 1st message, compute 2nd message	sign the
Auth patient to pharmacist (inject)	×	replay attack	add ch
Auth patient to pharmacist (non-inject)	√		

Table 1. Verification results of security and authentication of patient and doctor properties of the DLVV08 protocol.

(Strong) patient and doctor anonymity The DLVV08 protocol claims that no party should be able to determine the identity of a patient[6]. In this paper, not only the identity of a patient but also the identify of a doctor is considered. Earlier, it is verified that a patient's identity and a doctor's identity are not revealed in the protocol. In some cases, the adversary can distinguish a process initiated by one user from a process initiated by another user, although the users' identities are not revealed. This reduces the anonymous group. If the adversary can distinguish one user from the rest, i.e., the anonymous group is only one, then the user's privacy is revealed to some extend. Anonymity requires that the chance of a user executing a process is the same as other users executing the process.

Patient and doctor anonymity Doctor anonymity is defined as in Definition 7. To verify the property, is to check the satisfaction of the equivalence relation in the defintion. We model an equivalence as a bi-processes, which can be verified automatically in ProVerif. A bi-process models two processes which have the same structure, only differ on messages. The two processes are written as one process. The different parts of the two processes are modelled using $\mathsf{choice}[x, y]$,

[6] Notice patient anonymity is claimed as patient untraceability in the DLVV08 paper.

which means in the first process using x to replace $\mathsf{choice}[x, y]$ while using y to replace $\mathsf{choice}[x, y]$. For example, to verify the anonymity of doctor, we model two processes, the first one is initiated by $\mathtt{A}$, the second one is initiated by $\mathtt{B}$.This can be verified using bi-processes using ProVerif. The bi-process for doctor anonymity is as follows:

$\nu\tilde{m}.init.(!R_{pt}\ |!R_{dr}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ |\ (\nu\mathtt{Pnym}_{dr}; \text{let } \mathtt{Id}_{dr} = \mathsf{choice}[\mathtt{A}, \mathtt{B}] \text{ in } !P_{dr})$

Similar to doctor anonymity, patient anonymity is defined as

$$\mathcal{C}[init_{pt}\{\mathtt{A}/Id_{pt}\}.!P_{pt}\{\mathtt{A}/Id_{pt}\}] \approx_\ell \mathcal{C}[init_{pt}\{\mathtt{B}/Id_{pt}\}.!P_{pt}\{\mathtt{B}/Id_{pt}\}].$$

This can be verified in ProVerif using the following bi-process:

$$\begin{array}{ll} \nu\tilde{m}.init.(!R_{pt}\ |!R_{dr}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ | & (\text{ let } \mathtt{Id}_{pt} = \mathsf{choice}[\mathtt{A}, \mathtt{B}] \text{ in} \\ & \nu\mathtt{Pnym}_{pt}; \nu\mathtt{Sss}; \nu\mathtt{Acc}; \\ & \mathsf{in}(\mathtt{ch}_{hp}, \mathtt{Hii}); \\ & \text{let } c_{pt}\text{-}pk_{hii} = \mathsf{key}(\mathtt{Hii}) \text{ in } !P_{pt}) \end{array}$$

Strong patient and doctor anonymity Strong doctor anonymity is defined as in Definition 8. This definition for DLVV08 protocol is as follows:

$$DLV \approx_\ell \nu\tilde{m}.init.(!R_{pt}\ |!R_{dr}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ |\ R'_{dr}),$$

where

$$R'_{dr}{:=}\nu\mathtt{nPnym}_{dr}; \text{let } \mathtt{Pnym}_{dr} = \mathtt{nPnym}_{dr} \text{ in } !(P_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}\}),$$

in which $\mathtt{B}$ is a public doctor identity known by the adversary and never appears in other processes of the equivalences. Process R'_{dr} represents a doctor process, the initiator of which is known to the adversary.

Similarly, the definition of strong patient anonymity for DLVV08 protocol is as follows:

$$DLV \approx_\ell \nu\tilde{m}.init.(!R_{pt}\ |!R_{dr}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ |\ R'_{pt}),$$

where

$$\begin{array}{ll} R'_{pt} := & \nu\mathtt{Pnym}_{pt}; \nu\mathtt{Sss}; \nu\mathtt{Acc}; \\ & \mathsf{in}(\mathtt{ch}_{hp}, \mathtt{Hii}); \text{let } c_{pt}\text{-}pk_{hii} = \mathsf{key}(\mathtt{Hii}) \text{ in} \\ & !(P_{pt}\{\mathtt{B}/\mathtt{Id}_{pt}\}), \end{array}$$

in which $\mathtt{B}$ is a public patient identity known by the adversary and never appears in other processes of the equivalences. Process R'_{pt} represents a patient process, the initiator of which is known to the adversary.

To verify the anonymity of doctor, we model two processes, the first one models a normal protocol, the second one contains an additional process in which the doctor identity is a free name (meaning known by the adversary).This can be verified using bi-processes using ProVerif. The bi-process for verifying strong doctor anonymity is as

$$\begin{array}{ll} \multicolumn{2}{l}{\textit{free } \mathtt{B};} \\ \nu\tilde{m}.init.(& !R_{pt}\ |!R_{dr}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ | \\ & (\nu\mathtt{A}; \nu\mathtt{nPnym}_{dr}; \text{let } \mathtt{Pnym}_{dr} = \mathtt{nPnym}_{dr} \text{ in} \\ & !(\text{let } \mathtt{Id}_{dr} = \mathsf{choice}[\mathtt{A}, \mathtt{B}] \text{ in } P_{dr}))) \end{array}$$

Similarly, the bi-process for verifying patient anonymity is written as

$$
\begin{array}{l}
\textit{free}\ \mathtt{B};\\
\nu\tilde{m}.init.(\ !R_{pt}\ |!R_{dr}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ |\\
\qquad (\nu\mathtt{A};\nu\mathtt{Pnym}_{pt};\nu\mathtt{Sss};\nu\mathtt{Acc};\\
\qquad \mathsf{in}(\mathsf{ch}_{hp},\mathtt{Hii});\text{let }c_{pt}\text{-}pk_{hii}=\mathsf{key}(\mathtt{Hii})\text{ in}\\
\qquad !(\text{let }\mathtt{Id}_{pt}=\mathsf{choice}[\mathtt{A},\mathtt{B}]\text{ in }P_{pt})))
\end{array}
$$

Verification result. The verification shows that patient anonymity and strong patient anonymity are satisfied and doctor anonymity nor strong doctor anonymity is not satisfied.

The verification of strong doctor anonymity fails because the adversary can distinguish a process initiated by an unknown doctor and a known doctor. The way to distinguish them are as follows: given a doctor process, in which the doctor's identity is $\mathtt{A}$ and the doctor's pseudonym is $\mathtt{Pnym}_{dr}$, thus the doctor credential is $\mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{A})$, since $\mathtt{Pnym}_{dr}$ and $\mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{A})$ are revealed, the adversary knows them. The adversary also knows a public doctor identity $\mathtt{B}$, thus the adversary can fake a zero-knowledge proof $\mathsf{zk}((\mathtt{Pnym}_{dr},\mathtt{B}),\mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{A}))$. If the zero-knowledge proof passes the corresponding verification $\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{dr}}$ by the patient, the doctor process is executed by the doctor $\mathtt{B}$, otherwise, not. Thus, the adversary can tell the initiator of a process. Intuitively, when processes have different doctor pseudonyms are initiated by different doctors.

For the same reason, doctor anonymity fails. Both of them can be fixed using the assumption **s4'**.

(Strong) patient and doctor untraceability Even a user's identity is not revealed, the adversary may be able to distinguish whether two execution of the protocol is executed by the same user, thus traces the behaviour of a user. The DLVV08 protocol claims that prescriptions issued to the same patient should not be linkable to each other. In other words, the situation in which a patient executes the protocol twice should be indistinguishable with the situation in which two different patients execute the protocol individually. To satisfy this requirement, patient untraceability[7] is required.

Patient and doctor untraceability As in Definition 9, doctor untraceabiltiy can be verified in ProVerif using the bi-process:

$$
\begin{array}{l}
\nu\tilde{m}.init.(!R_{pt}\ |!R_{dr}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ |\\
(\ \nu\mathtt{nPnym}_{dr};\nu\mathtt{wPnym}_{dr};\\
\quad ((\text{let }\mathtt{Id}_{dr}=\mathtt{A}\text{ in let }\mathtt{Pnym}_{dr}=\mathtt{nPnym}_{dr}\text{ in }P_{dr})\ |\\
\quad (\text{let }\mathtt{Id}_{dr}=\mathsf{choice}[\mathtt{A},\mathtt{B}]\text{ in let }\mathtt{Pnym}_{dr}=\mathsf{choice}[\mathtt{nPnym}_{dr},\mathtt{wPnym}_{dr}]\text{ in }P_{dr}))))
\end{array}
$$

[7] Notice that patient untraceablility is claimed as patient unlinkability in DLVV08 paper, we use different terminology.

Similar to the definition of doctor untraceability, patient untraceability is defined as

$$\begin{aligned}&\mathcal{C}[\mathit{init}_{pt}\{\mathtt{A}/\mathit{Id}_{pt}\}.(P_{dr}\{\mathtt{A}/\mathit{Id}_{pt}\} \mid P_{dr}\{\mathtt{A}/\mathit{Id}_{pt}\})]\\ \approx_\ell\ &\mathcal{C}[(\mathit{init}_{pt}\{\mathtt{A}/\mathit{Id}_{pt}\}.P_{dr}\{\mathtt{A}/\mathit{Id}_{pt}\}) \mid (\mathit{init}_{pt}\{\mathtt{B}/\mathit{Id}_{pt}\}.P_{dr}\{\mathtt{B}/\mathit{Id}_{pt}\})].\end{aligned}$$

It is verified in ProVerif as the following bi-process:

$$\begin{array}{l}\nu\tilde{m}.\mathit{init}.(!R_{pt} \mid !R_{dr} \mid !R_{ph} \mid !R_{mpa} \mid !R_{hii} \mid\\ (\ \nu\mathtt{nPnym}_{pt};\nu\mathtt{nSss};\nu nAcc;\nu\mathtt{wPnym}_{pt};\nu\mathtt{wSss};\nu wAcc;\\ \quad \mathsf{in}(\mathsf{ch}_{hp}, nHii);\mathsf{in}(\mathsf{ch}_{hp}, wHii);\\ \quad \text{let } c_{pt}\_npk_{hii} = \mathsf{key}(nHii) \text{ in}\\ \quad \text{let } c_{pt}\_wpk_{hii} = \mathsf{key}(wHii) \text{ in}\\ \quad (\text{let } \mathtt{Hii} = nHii \text{ in let } c_{pt}\_pk_{hii} = c_{pt}\_npk_{hii} \text{ in let } \mathtt{Id}_{pt} = \mathtt{A} \text{ in}\\ \quad\ \text{let } \mathtt{Pnym}_{pt} = \mathtt{nPnym}_{pt} \text{ in let } \mathtt{Sss} = \mathtt{nSss} \text{ in let } \mathtt{Acc} = nAcc \text{ in } P_{pt}) \mid\\ \quad (\text{let } \mathtt{Hii} = \mathsf{choice}[nHii, wHii] \text{ in let } c_{pt}\_pk_{hii} = \mathsf{choice}[c_{pt}\_npk_{hii}, c_{pt}\_wpk_{hii}] \text{ in}\\ \quad\ \text{let } \mathtt{Id}_{pt} = \mathsf{choice}[\mathtt{A},\mathtt{B}] \text{ in let } \mathtt{Pnym}_{pt} = \mathsf{choice}[\mathtt{nPnym}_{pt}, \mathtt{wPnym}_{pt}] \text{ in}\\ \quad\ \text{let } \mathtt{Sss} = \mathsf{choice}[\mathtt{nSss}, \mathtt{wSss}] \text{ in let } \mathtt{Acc} = \mathsf{choice}[nAcc, wAcc] \text{ in } P_{pt})))\end{array}$$

Strong patient and doctor untraceability Strong untraceability is modelled as a patient executing the protocol repeatedly is indistinguishable to different patients executing the protocol each once. Strong doctor untraceability is defined as in Definition 10. Strong doctor untraceability is modelled for DLVV08 protocol as

$$DLV \approx_\ell \nu\tilde{m}.\mathit{init}.(!R_{pt} \mid !R'_{dr} \mid !R_{ph} \mid !R_{mpa} \mid !R_{hii}),$$

where

$$R'_{dr} := \nu\mathtt{Id}_{dr};\nu\mathtt{Pnym}_{dr};P_{dr}$$

Similarly strong patient untraceability for DLVV08 protocol is modelled as:

$$DLV \approx_\ell \nu\tilde{m}.\mathit{init}.(!R'_{pt} \mid !R_{dr} \mid !R_{ph} \mid !R_{mpa} \mid !R_{hii}),$$

where

$$\begin{array}{l}R'_{pt} :=\\ \qquad \nu\mathtt{Id}_{pt};\nu\mathtt{Pnym}_{pt};\nu\mathtt{Sss};\nu\mathtt{Acc};\\ \qquad \mathsf{in}(\mathsf{ch}_{hp}, \mathtt{Hii});\\ \qquad \text{let } c_{pt}\_pk_{hii} = \mathsf{key}(\mathtt{Hii}) \text{ in}\\ \qquad (\mathsf{in}(\mathsf{ch}_{phpt}, rcv\_pk_{ph});\\ \qquad \text{let } rcv_{pt}\_pk_{ph} = rcv\_pk_{ph} \text{ in}\\ \qquad \text{let } \mathtt{Id}_{ph} = \mathsf{host}(rcv\_pk_{ph}) \text{ in } P_{pt})\end{array}$$

To verify untraceability of patients and doctors, we model the equivalences as bi-processes in ProVerif. The bi-process for verifying patient untraceability is modelled as

$$
\begin{aligned}
\nu\tilde{m}.init.(\ & !R_{dr}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ | \\
& !(\nu\mathtt{nId}_{pt};\nu\mathtt{nPnym}_{pt};\nu\mathtt{nSss};\nu nAcc; \\
& \mathsf{in}(\mathtt{ch}_{hp}, nHii); \\
& !(\nu\mathtt{wId}_{pt};\nu\mathtt{wPnym}_{pt};\nu\mathtt{wSss};\nu wAcc; \\
& \text{let } \mathtt{Id}_{pt} = \mathsf{choice}[\mathtt{nId}_{pt}, \mathtt{wId}_{pt}] \text{ in} \\
& \text{let } \mathtt{Pnym}_{pt} = \mathsf{choice}[\mathtt{nPnym}_{pt}, \mathtt{wPnym}_{pt}] \text{ in} \\
& \text{let } \mathtt{Sss} = \mathsf{choice}[\mathtt{nSss}, \mathtt{wSss}] \text{ in} \\
& \text{let } \mathtt{Acc} = \mathsf{choice}[nAcc, \mathtt{wSss}] \text{ in} \\
& \mathsf{in}(\mathtt{ch}_{hp}, wHii); \\
& \text{let } \mathtt{Hii} = \mathsf{choice}[nHii, wHii] \text{ in} \\
& \text{let } c_{pt}\text{-}pk_{hii} = \mathsf{key}(\mathtt{Hii}) \text{ in} \\
& P_{pt})))
\end{aligned}
$$

Similarly, the bi-process for verifying patient untraceability is modelled as:

$$
\begin{aligned}
\nu\tilde{m}.init.(!R_{pt}\ |!R_{ph}\ |!R_{mpa}\ |!R_{hii}\ |\ & !(\nu\mathtt{nId}_{dr};\nu\mathtt{nPnym}_{dr}; \\
& !(\nu\mathtt{wId}_{dr};\nu\mathtt{wPnym}_{dr}; \\
& \text{let } \mathtt{Id}_{dr} = \mathsf{choice}[\mathtt{nId}_{dr}, \mathtt{wId}_{dr}] \text{ in} \\
& \text{let } \mathtt{Pnym}_{dr} = \mathsf{choice}[\mathtt{nPnym}_{dr}, \mathtt{wPnym}_{dr}] \text{ in } P_{dr})))
\end{aligned}
$$

Verification result. Both doctor untraceability and patient untraceability fail the verification.

Strong doctor untraceability fails because the adversary can distinguish two processes initiated by one doctor or by two doctors. Since the doctor's pseudonym is revealed in the DLVV08 protocol and we assume a doctor uses the same pseudonym in all sessions, when two doctor pseudonyms are the same, the two processes are initiated by the same doctor, otherwise, not. Doctor untraceability fails for the same reason. Both of them can be fixed by revising the assumption **s3'**.

Strong patient untraceability fails. The adversary can distinguish two processes initiated by two patients and by one patient, because of the following reasons. First, the social security status is revealed, and we assume that a patient uses the same status in all sessions and different patients have different status. Therefore, the adversary distinguish two processes initiated by one patient (with two identical statuses) and initiated by two patients (with two different statuses), thus, the adversary can trace a patient. Second, we assume a patient uses the same patient pseudonym and the same patient HII in all sessions. Since the public key of social safety organisation is public information, the adversary can distinguish two process initiated by one patient (with two identical cipher texts $\mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{sso})$ and two indentical $\mathsf{enc}(\mathtt{Hii}, \mathtt{pk}_{sso})$) and initiated by two patients (with two different cipher texts $\mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{sso})$ and two different $\mathsf{enc}(\mathtt{Hii}, \mathtt{pk}_{sso})$). Thus, the adversary can trace a patient by comparing the encryptions ($\mathsf{enc}(\mathtt{Pnym}_{pt}, \mathtt{pk}_{sso})$, and $\mathsf{enc}(\mathtt{Hii}, \mathtt{pk}_{sso})$). Third, the patient credential is not freshly generated, and we assume a patient uses the same credential in all sessions. The adversary can trace a patient by the patient's credential.

Fourth, the adversary can distinguish two processes using the same HII and two processes using different HIIs. A patient is traceable if all other patients have different HIIs. Patient untraceability fails for the same reason. Both of them can be fixed by applying assumptions revising the assumptions (**s5'**, **s2'**, **s4"** and **s6'**).

Prescribing-privacy The definition of prescribing-privacy is defined in Section 3. To verify the prescribing-privacy is to check the satisfaction of the equivalence in the definition:

$$\begin{aligned} &\mathcal{C}[(init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{a}/presc\})) \mid \\ &\quad (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{b}/presc\}))] \\ \approx_{\ell} &\mathcal{C}[(init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{b}/presc\})) \mid \\ &\quad (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{a}/presc\}))]. \end{aligned}$$

The context $\mathcal{C}$ in the DLVV08 protocol is as follows,

$$\mathcal{C} = \nu\tilde{m}.init.(!R_{pt} \mid !R_{dr} \mid !R_{ph} \mid !R_{mpa} \mid !R_{hii} \mid \_).$$

Thus, the equivalence is modelled as a bi-process in ProVerif, as follows:

```
$free$ A;
$free$ B;
$free$ a;
$\nu\tilde{m}.init.($ $!R_{pt}$ $|!R_{dr}$ $|!R_{ph}$ $|!R_{mpa}$ $|!R_{hii}$ $|$
        $(\nu$nPnym$_{dr}$; $\nu$wPnym$_{dr}$;
        let Id$_{dr}$ = choice[A, B] in
        let Pnym$_{dr}$ = choice[nPnym$_{dr}$, wPnym$_{dr}$] in
        let presc = a in $main_{dr}) \mid$
        $(\nu$nPnym$_{dr}$; $\nu$wPnym$_{dr}$;
        let Id$_{dr}$ = choice[B, A] in
        let Pnym$_{dr}$ = choice[nPnym$_{dr}$, wPnym$_{dr}$] in
        let presc = b in $main_{dr}))$
```

Verification result. The verification results shows that prescribing-privacy is not satisfied, for the reason that the adversary can distinguish a prescription is prescribed by doctor A or doctor B. Since we assume two doctors have two different identities (public) and two different pseudonyms (originally private, but published later), two doctors have different credentials. By comparing the credential related to the prescription, the adversary can tell which doctor prescribed it. This can be fixed by revising the assumption **s4'**.

7.2 Analysis of enforced privacy properties

With the original assumption, the DLVV08 protocol does not satisfy prescribing-privacy. We conjecture that independency of enforced prescribing-privacy implies independency of prescribing-privacy and enforced prescribing-privacy, each of which also implies prescribing-privacy. Thus, the protocol does not satisfies enforced prescribing-privacy, independency of prescribing-privacyand independency of enforced prescribing-privacy. Therefore, when we talk about verification of enforced prescribing-privacy, independency of prescribing-privacyand independency of enforced prescribing-privacyin this section (Section 7.2), we refer it as the DLVV08 protocol with assumption **s4'**.

Enforced prescribing-privacy Enforced prescribing-privacymeans a doctor cannot prove his prescription to the adversary. Enforced prescribing-privacyhas two assumptions: first, the doctor reveals all his private information to the adversary, second, the adversary does not interfere the protocol. The definition of enforced prescribing-privacy, in Section 3.2, is modelled as the existence of a process P'_{dr}, such that the following two equivalences are satisfied,

$$\begin{aligned}
&\mathcal{C}[(\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid P'_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\})) \mid \\
&\quad (\mathit{init}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}, \mathtt{a}/\mathit{presc}\}))] \\
\approx_\ell\ &\mathcal{C}[((\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\})^{\mathtt{chc}}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid (\mathit{main}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}, \mathtt{a}/\mathit{presc}\})^{\mathtt{chc}})) \mid \\
&\quad (\mathit{init}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}\}.(!\mathit{main}_{dr}\{\mathtt{B}/\mathit{Id}_{dr}, \mathtt{b}/\mathit{presc}\}))], \\
\mathit{and}, \\
&(\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid P'_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}^{\backslash \mathtt{out}(\mathtt{chc},\cdot)})) \\
\approx_\ell\ &(\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}, \mathtt{b}/\mathit{presc}\})).
\end{aligned}$$

Due to the existence in the definition, we cannot verify enforced prescribing-privacy directly using ProVerif. We can use ProVerif to verify equivalences, but cannot use it to prove the existence of a process.

However, we find an intuitive flaw in the DLVV08 protocol which shows that the protocol does not satisfy enforced prescribing-privacy. A doctor is able to prove to the adversary of his prescription using the following steps:

- A doctor communicates with the adversary (e.g., a pharmaceutical company) to agree with a bit-commitment the doctor is going to use. Therefore, it links a doctor and the bit-commitments.
- The doctor uses the agreed bit-commitment in the communication between the doctor and the patient. Therefore, it links the bit-commitment to a prescription.
- Later, when the patient uses this prescription to get medicine from the pharmacist, the adversary can observe the prescription being used. Therefore, it proves the doctor really prescript the medicine. The pharmaceutical company can pay the doctor.

Formally, we show, using ProVerif, that when a doctor reveals his information to the adversary, prescribing-privacy is broken. To prove that there is no alternative precesses for a doctor to cheat the adversary, we assume there exists a process P'_{dr} which satisfies the definition of enforced prescribing-privacy, and find contradictions.

Proof. Assume there exists a process P'_{dr} which satisfies the definition of enforced prescribing-privacy, i.e.,

$$\begin{aligned}
&\mathcal{C}[(\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\} \mid P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\})) \mid \\
&\quad (\mathit{init}_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}\}.(!P_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}, \mathtt{a}/\mathtt{presc}\}))] \\
\approx_\ell\ &\mathcal{C}[((\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\})^{\mathtt{chc}}.(!P_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\} \mid (\mathit{main}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{a}/\mathtt{presc}\})^{\mathtt{chc}}))],
\end{aligned} \tag{eq1}$$

and,

$$\begin{aligned}
&(\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}.(P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}^{\setminus \mathsf{out}(ch,\cdot)})) \\
\approx_\ell\ &(\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}.(\mathit{main}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{b}/\mathtt{presc}\})).
\end{aligned} \tag{eq2}$$

For the first equivalence $eq1$, if $M =_E N$ on the left hand side, then $M =_E N$ on the right hand side.

On the right hand side of $eq1$, there exists an output of a prescription proof $\mathit{PrescProof}^{\mathtt{r}}$, from process

$$\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\})^{\mathtt{chc}}.(!P_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\} \mid (\mathit{main}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{a}/\mathtt{presc}\})^{\mathtt{chc}}).$$

The adversary can obtain the prescription and the doctor commitment from the prescription proof.

$$(\mathsf{a}, \mathit{PrescriptID}^r, \mathit{Comt}_{dr}, \mathit{c\_Comt}_{pt}{}^r) = \mathsf{getSpkMsg}(\mathit{PrescProof}^{\mathtt{r}})$$

On the left hand side of $eq1$, there should also exists an output of a prescription proof $\mathit{PrescProof}$ from which the adversary can obtain a prescription $\mathtt{a}$ and a doctor commitment Comt_{dr}.

On the right hand side, there is

$$\mathit{Comt}_{dr} = \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{nonce})$$

where $\mathtt{Pnym}_{dr}$ and $\mathtt{nonce}$ are revealed to the adversary on $\mathtt{chc}$ channel. On the left hand side, the only process which can output messages on $\mathtt{chc}$ channel is $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$. Thus, the only process which can generate the doctor commitment Comt_{dr} on the left hand side is $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$. Thus, the process which outputs the prescription proof $\mathit{PrescProof}$ on the left hand side is $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$. Thus, the prescription in process $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$ is a.

However, on the right hand side of the second equivalence $eq2$, there is a prescription proof output from process $\mathit{main}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{b}/\mathtt{presc}\}$. The adversary obtains prescription $\mathtt{b}$ from the prescription proof. To satisfy equivalence $eq2$, the process $(P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}^{\setminus \mathsf{out}(ch,\cdot)})$ should also output a prescription proof $\mathit{PrescProof}'$ where the prescription needs to be $\mathtt{b}$. Since $(P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}^{\setminus \mathsf{out}(ch,\cdot)})$ outputs $\mathit{PrescProof}'$, $(P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$ should also output $\mathit{PrescProof}'$. There should only be one prescription proof observable to the adversary from process $(P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$. Thus, $\mathit{PrescProof}' = \mathit{PrescProof}$. Thus, $\mathtt{b} = \mathsf{a}$.

Intuitively, a doctor cannot lie about a prescription and his pseudonym, since they are public information. The only thing the doctor may be able to lie about is the link of doctor and his prescription. Since the link between the commitment of doctor pseudonym and the prescription is obvious, if the doctor tells the adversary that he prescribed $\mathtt{a}$, while the adversary observes that the bit-commitments are linked to $\mathtt{B}$. The adversary can tell the $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\})$ from $(main_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{a}/\mathtt{presc}\})^{\mathtt{chc}}))$. Therefore there does not exist such process $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\})$.

Prescribing-privacy independent of pharmacist The DLVV08 protocol claims that pharmacists should not be able to provide evidence to pharmaceutical companies about doctor's prescription behaviour. This requirement is captured by the property prescribing-privacy independent of pharmacist. Following the definition of independency of prescribing-privacyin Definition 5, the prescribing-privacy independent of pharmacist of the DLVV08 protocol is defined as follows:

$$\begin{array}{rl}
\mathcal{C}[!{R_{ph}}^{\mathtt{ch}} \mid & (init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{a}/presc\})) \mid \\
& (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{b}/presc\}))] \\
\approx_\ell \mathcal{C}[!{R_{ph}}^{\mathtt{ch}} \mid & (init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{b}/presc\})) \mid \\
& (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{a}/presc\}))],
\end{array}$$

where the context $\mathcal{C}$ is defined as

$$\mathcal{C} = \nu\tilde{m}.init.(!R_{pt} \mid !R_{dr} \mid !R_{ph} \mid !R_{mpa} \mid !R_{hii} \mid \_).$$

To verify the equivalence, we verify the bi-process

$$\begin{array}{ll}
\nu\tilde{m}.init.(& !R_{pt} \mid !R_{mpa} \mid !R_{hii} \mid !R_{ph} \mid !(R_{ph})^{\mathtt{chc}} \mid \\
& (\nu \mathtt{nPnym}_{dr}; \nu \mathtt{wPnym}_{dr}; \\
& \text{let } \mathtt{Id}_{dr} = \mathsf{choice}[\mathtt{A}, \mathtt{B}] \text{ in} \\
& \text{let } \mathtt{Pnym}_{dr} = \mathsf{choice}[\mathtt{nPnym}_{dr}, \mathtt{wPnym}_{dr}] \text{ in} \\
& \text{let } \mathtt{presc} = \mathtt{a} \text{ in } main_{dr}) \mid \\
& (\nu \mathtt{nPnym}_{dr}; \nu \mathtt{wPnym}_{dr}; \\
& \text{let } \mathtt{Id}_{dr} = \mathsf{choice}[\mathtt{B}, \mathtt{A}] \text{ in} \\
& \text{let } \mathtt{Pnym}_{dr} = \mathsf{choice}[\mathtt{nPnym}_{dr}, \mathtt{wPnym}_{dr}] \text{ in} \\
& \text{let } \mathtt{presc} = \mathtt{b} \text{ in } main_{dr}))
\end{array}$$

The verification result shows that the protocol satisfies pharmacist-independent doctor-privacy.

Pharmacist independency of enforced prescribing-privacy The DLVV08 protocol intends to prevent bribery between doctors and pharmaceutical companies and prevent pharmacists revealing information to harm a doctor. This requirement is captured by the property pharmacist-independent enforced prescribing-privacy. Following the definition of independency of enforced prescribing-privacyin Definition 6, enforced prescribing-privacy independent of pharmacist is defined as the existence of a process P'_{dr}, such that the following equivalences are satisfied:

$$\begin{aligned}
&\mathcal{C}[!(R_{ph})^{\mathsf{chc}} \mid ((init_{dr}\{\mathtt{A}/Id_{dr}\})^{\mathsf{chc}}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid P'_{dr}\{\mathtt{A}/Id_{dr}\})) \mid \\
&\quad (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!P_{dr}\{\mathtt{B}/Id_{dr}\} \mid main_{dr}\{\mathtt{B}/Id_{dr}, a/presc\}))] \\
\approx_\ell\ &\mathcal{C}[!(R_{ph})^{\mathsf{chc}} \mid ((init_{dr}\{\mathtt{A}/Id_{dr}\})^{\mathsf{chc}}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid (main_{dr}\{\mathtt{A}/Id_{dr}, a/presc\})^{\mathsf{chc}})) \mid \\
&\quad (init_{dr}\{\mathtt{B}/Id_{dr}\}.(!main_{dr}\{\mathtt{B}/Id_{dr}, \mathtt{b}/presc\}))],
\end{aligned}$$

and,

$$\begin{aligned}
&(init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid P'_{dr}\{\mathtt{A}/Id_{dr}\}^{\backslash \mathsf{out}(ch,\cdot)})) \\
\approx_\ell\ &(init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathtt{b}/presc\})).
\end{aligned}$$

Same as in enforced prescribing-privacy, we cannot proof enforced prescribing-privacy independent of pharmacist using ProVerif because of the existence quantification in the definition. However, the DLVV08 protocol can be verified by finding a flaw which shows that the protocol does not satisfy pharmacist-independent enforced prescribing-privacy. One intuitive flaw is the same flaw in enforced prescribing-privacy. Intuitively, when a doctor can prove his prescription without pharmacist showing information to the adversary, the doctor can prove it with pharmacist showing information to the adversary, given that the pharmacist genuinely cooperate with the adversary. Since the protocol does not satisfies enforced prescribing-privacy, it does not satisfies enforced prescribing-privacyindependent of pharmacists.

checked privacy property	**initial model**	**cause(s)**	**improvement**	**revised model**
prescribing-privacy	×	**s4**	**s4'**	√
enforced presc.-priv.	× (with **s4'**)		**s8'**	√
independency of presc.-priv.	√ (with **s4'**)			√
independency of enforced presc.-priv.	×(with **s4'**)		**s8'**	×
patient anonymity	√			√
strong patient anonymity	√			√
doctor anonymity	×	**s4**	**s4'**	√
strong doctor anonymity	×	**s4**	**s4'**	√
patient untraceability	×	**s2**, **s4**, **s5**, **s6**	**s2'**, **s4"**, **s5'**, **s6'**	√
strong patient untraceability	×	**s2**, **s4**, **s5**, **s6**	**s2'**, **s4"**, **s5'**, **s6'**	√
doctor untraceability	×	**s3**	**s3'**	√
strong doctor untraceability	×	**s3**	**s3'**	√

Table 2. Verification results of the DLVV08 protocol with original and revised assumptions.

7.3 Other flaws

In the protocol, if sending invoice or sending reception is blocked, the pharmacists won't get the reimbursement. Therefore the channels used in the protocol needs to be unblockable channels.

8 Addressing the flaws of the DLVV08 protocol

In order to guarantee that the DLVV08 protocol satisfies the above mentioned properties, we propose some suggestions for fixing them.

8.1 Addressing the flaws of regular security and privacy properties

Fix secrecy. A patient's social security status and a doctor's pseudonym do not satisfy secrecy respecting to Dolev-Yao adversary. A patient's social security status is revealed because of the message which intends to prove the social security status to the doctor. To fix secrecy of a patient's social security status, it requires that prove only reveals the social security status to the pharmacist. Since how a social security status is represented and what the pharmacist needs to verify, are not clear, we cannot give explicit suggestions. For example, if the social security status is a number, and the pharmacist only needs to verify that the number is higher than a certain threshold, the patient can prove it using zero-knowledge proof without revealing the number; if the pharmacist needs to verify the exact value of the status, one way to fix its secrecy is that the pharmacist and the patient agree on a session key and the status is encrypted using the key.

A doctor's pseudonym is revealed because of the doctor commited it and later sends the open information. One way to fix it is that the open information is encrypted using an agreed session key.

Fix authentications. A doctor cannot authenticate a patient. To fix it, one way is to add a challenge from the doctor, when the patient authenticate to the doctor, the patient needs to include the challenge in the proof. It assures that the proof is freshly generated. Thus it prevents relaying of old messages.

Fix doctor anonymity. The essential reason that the (strong) doctor anonymity is not satisfied, is that the adversary can fake an anonymous authentication of a doctor. To fix the doctor anonymity, one way is that a doctor uses a freshly generated doctor credential in each session and proves that he knows the randomness **s4'**. We model it as adding a random number to the credential parameters. To ensure the adversary cannot replay an old credential, we modify the anonymous authentication proof by adding a proof that the doctor knows the random number. The intuition is that by adding randomness, the adversary cannot apply dictionary guess to the doctor identity.

Fix doctor untraceability. The essential reason that the (strong) doctor untraceability is not satisfied, is that a doctor's pseudonym is revealed and is used for all sessions. One way to fix doctor untraceablity is to make sure that a doctor's pseudonym is freshly generated in each session **s3'**.

Fix patient untraceability. The essential reasons that the (strong) patient untraceability is not satisfied, are that 1) a patient's social security status is revealed and is the same in all sessions; 2) a patient's pseudonym and HII are the same in all sessions, thus, the encryptions of them are the same in all sessions; 3) a patient's credential is the same in all sessions; 4) a patient's HII is the same in all sessions. There are several ways to fix the first problem, for example, make sure the social security status is not revealed, the social security status is different in each session, or any two patients have the same social security status. It is not clear what exactly the status is and how many types of statuses there are, we choose to assume that a patient's social security status is different in each session **s5'**. If we assume two patients have the same social security status, it reduces the untraceability of a patient to only among those who share the same status. To fix the second problem, one way is that the encryption to be probability encryption **s2'**, thus, even a patient uses the same pseudonym in different sessions, the patient is not traceable because of the cipher texts. To fix the third problem, one way is to make sure that the patient credential to be freshly generated **s4"**. To fix the fourth problem, we refer to patient untraceabbility with respect to those who share the same HII **s6'**. It is not realistic that a patient changes his HII every time he executes the protocol.

Update the assumptions to satisfy privacy properties. There are several ways to fix each flaw which is found during verifying security and privacy properties. We choose one solution for fixing each flaw and update the protocol. Note that we do not fix secrecy and authentication problems, because it is clear to what extend of secrecy and authentication the protocol should satisfy.

The following lists the changes in the assumptions:

s2' The encryptions are probabilistic.

s3' A doctor's pseudonyms are freshly generated for each execution.

s4' A doctor's credential and anonymous authentication are freshly generated for each execution.

s4" A patient's credential and anonymous authentication are freshly generated for each execution.

s5' A patient's social security status changes in each execution.

s6' All patients share the same HII. Each HII has large amount of patients, such that patients of the same HII form a group, hiding a particular patient. A patient is untraceable only respect to the group. A patient's HII must be a trust party, which does not reveal the link between the patient's pseudonym and the patient's identity.

s8" The channels used in the protocol are unblockable.

We verified the following privacy properties: (strong) doctor and patient anonymity, (strong) doctor and patient untraceability, prescribing-privacy, of the protocol with the updated assumptions. The way to verify each property is the same as described in the previous section. The results show that the protocol with updated assumptions satisfies the properties (doctor and patient anonymity, doctor and patient untraceability, prescribing-privacy).

8.2 Fix enforced prescribing-privacy.

Try chameleon bit-commitments To fix the enforced prescribing-privacy, we borrow the solution from voting, using chameleon bit-commitments to hide the prescription. This solution assumes that there is an untappable channel built between a patient and a doctor, as well as between a patient and a pharmacist. The untappable channels are used to send the random number used in chameleon bit-commitments. The model of the protocol with chameleon bit-commitments is shown in Section D in Appendix.

However, we did not find a way for a bribed doctor to cheat, because the adversary can always detect whether a bribed doctor lied. Suppose the adversary asks the bribed doctor to prescribe a, while the doctor prescribed b. The doctor reveals his commitment $\mathsf{ChCommit}(\mathsf{b}, r)$, lying to the adversary of the random number $\mathsf{fake}(\mathsf{ChCommit}(\mathsf{b}, r), \mathsf{a})$. The adversary can fake the prescription proof since the bribed doctor needs to reveal his pseudonym, identity and random numbers used. Thus the adversary can alter the prescription commitment to $\mathsf{ChCommit}(c, \mathsf{fake}(\mathsf{ChCommit}(\mathsf{b}, r), \mathsf{a}))$. When the pharmacist received this commitment and the real random number r, the pharmacist cannot open it. Thus, the adversary detect that the bribed doctor lied. Essentially, this solution does not work because the adversary can block and change a prescription proof.

We show using ProVerif that the most intuitive doctor process P'_{dr} in Figure 46 does not satisfy enforced prescribing-privacy.

Fix using untappable channels. We assume that after authentication of in each sub-protocol, the two communication parties establish an untappable channel. All other information are send through untappable channels.

s8' The communication channels are untappable, except that communication channels for authentications remain public.

We show that there exists a process P'_{dr}, which satisfies the two equivalences in the enforced prescribing-privacy definition 4. The equivalences are verified using ProVerif.

We also show that under this assumption (untappable channel), even without assumption **s4'**, i.e. with the original assumptions, the protocol still satisfies enforced prescribing-privacy.

However, the model which satisfies enforced prescribing-privacy does not satisfy prescribing-privacy independent of pharmacist .

The proof is similar to the proof of the dissatisfaction of enforced proscribing privacy. To prove that there is no alternative precesses for a doctor to cheat the adversary, we assume there exists a process P'_{dr} which satisfies the definition of independency of enforced prescribing-privacy, and then find contradictions.

Proof. Suppose, there exists a process P'_{dr} which satisfies the definition of independency of enforced prescribing-privacy, i.e.,

$$\begin{aligned}
&\mathcal{C}[!P_{ph}^{\mathsf{chc}} \mid (\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\} \mid P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\})) \mid \\
&\qquad\qquad (\mathit{init}_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}\}.(!P_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}, \mathtt{a}/\mathtt{presc}\}))] \\
\approx_\ell\ &\mathcal{C}[!P_{ph}^{\mathsf{chc}} \mid ((\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\})^{\mathsf{chc}}.(!P_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\} \mid (\mathit{main}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{a}/\mathtt{presc}\})^{\mathsf{chc}})) \\
&\qquad\qquad (\mathit{init}_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}\}.(!P_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{B}/\mathtt{Id}_{dr}, \mathtt{b}/\mathtt{presc}\}))]
\end{aligned} \tag{eq1}$$

and,

$$\begin{aligned}
&(\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}.(P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}^{\setminus \mathsf{out}(ch,\cdot)})) \\
\approx_\ell\ &(\mathit{init}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}.(\mathit{main}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{b}/\mathtt{presc}\}))
\end{aligned} \tag{eq2}$$

Because of the first equivalence $eq1$, if $M =_E N$ on the left hand side, then $M =_E N$ on the right hand side.

On the right hand side, there exists a prescription proof $\mathit{PrescProof}^{\mathtt{r}}$ in process $(\mathit{main}_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}, \mathtt{a}/\mathtt{presc}\})^{\mathsf{chc}}))$. This prescription proof eventually is revealed to the adversary on the $\mathtt{chc}$ channel by a pharmacist. The adversary can obtain a prescription a and a doctor commitment Comt_{dr} from it using

$$(\mathsf{a}, \mathit{PrescriptID}^r, \mathit{Comt}_{dr}, \mathit{c\_Comt_{pt}}^r) = \mathsf{getSpkMsg}(\mathit{PrescProof}^{\mathtt{r}}).$$

The adversary can also obtain the doctor credential Cred_{dr} and the same doctor commitment Comt_{dr} from the prescription proof using

$$(\mathit{Comt}_{dr}, \mathit{Cred}_{dr}) = \mathsf{getSpkVinfo}(\mathit{PrescProof}^{\mathtt{r}}).$$

On the left hand side, there should also exist an output of a prescription proof $\mathit{PrescProof}^l$ on the $\mathtt{chc}$ channel by a pharmacist, such that the adversary can obtain $\mathtt{a}$ and Comt_{dr}

$$(\mathsf{a}, \mathit{PrescriptID}^l, \mathit{Comt}_{dr}, \mathit{c\_Comt_{pt}}^l) = \mathsf{getSpkMsg}(\mathit{PrescProof}^l),$$

and obtain Comt_{dr} and Cred_{dr} by applying

$$(\mathit{Comt}_{dr}, \mathit{Cred}_{dr}) = \mathsf{getSpkVinfo}(\mathit{PrescProof}^l).$$

On the left hand side, there is

$$\mathit{Comt}_{dr} = \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{nonce})$$

where $\mathtt{Pnym}_{dr}$, $\mathtt{nonce}$ are revealed to the adversary on $\mathtt{chc}$ channel, and

$$\mathit{Cred}_{dr} = \mathsf{drcred}(\mathtt{A}, \mathtt{Pnym}_{dr}, \mathtt{n}_{dr})$$

where $\mathtt{Pnym}_{dr}$ and $\mathtt{n}_{dr}$ are revealed to the adversary on $\mathtt{chc}$ channel.

On the right hand side, the only process which can output on $\mathtt{chc}$ channel is $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$, thus, the process generating $\mathtt{nonce}$ $\mathtt{n}_{dr}$ is $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$. Thus, the process computing $\mathit{PrescProof}^l$ is $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$. Since the pharmacist received prescription is the prescription a doctor did prescribe, thus, the doctor initiated process $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$ prescribes a.

However, on the second equivalence $eq2$, on the right hand side the doctor prescribes b, thus, on the left hand side the process $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}^{\backslash \mathsf{out}(ch,\cdot)}$ generates b. Thus, in process $P'_{dr}\{\mathtt{A}/\mathtt{Id}_{dr}\}$ the prescription proof is $\mathit{PrescProof}'$ where the prescription is b.s

We assume pharmacists genuinely forwards all their information to the adversary. To satisfy the second equivalence, the pharmacist should output $\mathit{PrescProof}'$ (with b) on chc channel, while according to the first equivalence, the pharmacist should output $\mathit{PrescProof}^{l}$ (with a) on the chc channel.

Intuitively, all information sent over untappable channels are received by pharmacists and can be genuinely revealed to the adversary by the pharmacists (do not lie by assumption). Hence, there still exist links between a doctor, his nonces, his commitment, his credential and his prescription, when the doctor is bribed/coerced to reveal the nonces used in the commitment and the credential to the adversary. A doctor is linked to the nonce he used in his commitment. A doctor's commitment is linked to his prescription in the prescription proof. A doctor's prescription proof is sent over untappable channels first to a patient and later from the patient to a pharmacist, thus a malicious pharmacist can reveal the prescription proof to the adversary through a different channel (see Def. 6). If a bribed doctor lied about his prescription, the adversary can detect it by checking the doctor's corresponding prescription proof revealed by the pharmacist. The untappable channel assumption makes the protocol satisfy enforced prescribing-privacy while not satisfy independency of enforced prescribing-privacy because untappalbe channel enable a bribed doctor to lie and we assume pharmacist does not lie.

9 Conclusion

In this paper, we identify enforced privacy requirements in eHealth protocols, study enforced prescribing-privacy, define independency of prescribing-privacy, and independency of enforced prescribing-privacy. All these properties are formalised in the applied pi calculus. We take the DLVV08 protocol as a case study. We model the protocol in the applied pi calculus and analyse security and privacy properties of the protocol. We address ambiguities and flaws in the protocol and propose solutions for fixing them.

References

1. Oh, H., Rizo, C., Enkin, M., Jadad, A.: What is ehealth?: a systematic review of published definitions. World Hosp Health Serv **41**(1) (2005) 32–40
2. Louwerse, K.: The electronic patient record; the management of access – case study: Leiden University hospital. International Journal of Medical Informatics **49**(1) (1998) 39–44

3. Reid, J., Cheong, I., Henricksen, M., Smith, J.: A novel use of rBAC to protect privacy in distributed health care information systems. In: Proc. 8th Australian Conference on Information Security and Privacy. Volume 2727 of LNCS., Springer (2003) 403–415
4. Currim, F., Jung, E., Xiao, X., Jo, I.: Privacy policy enforcement for health information data access. In: Proc. 1st ACM Workshop on Medical-grade Wireless Networks, ACM Press (2009) 39–44
5. Dolev, D., Yao, A.C.C.: On the security of public key protocols. IEEE Transactions on Information Theory **29**(2) (1983) 198–207
6. Benaloh, J., Tuinstra, D.: Receipt-free secret-ballot elections (extended abstract). In: Proc. 26th Symposium on Theory of Computing, ACM Press (1994) 544–553
7. Lee, B., Kim, K.: Receipt-free electronic voting through collaboration of voter and honest verifier. In: Proc. Japan-Korea Joint Workshop on Information Security and Cryptology. (2000) 101–108
8. Hirt, M., Sako, K.: Efficient receipt-free voting based on homomorphic encryption. In: Proc. 19th Conference on the Theory and Application of Cryptographic Techniques. Volume 1807 of LNCS., Springer (2000) 539–556
9. Lee, B., Kim, K.: Receipt-free electronic voting with a tamper-resistant randomizer. In: Proc. 4th Conference on Information and Communications Security. Volume 2513 of LNCS., Springer (2002) 389–406
10. Decker, B., Layouni, M., Vangheluwe, H., Verslype, K.: A privacy-preserving ehealth protocol compliant with the belgian healthcare system. In: Proc. 5th European PKI workshop on Public Key Infrastructure. (2008) 118–133
11. Abadi, M., Fournet, C.: Mobile values, new names, and secure communication. In: Proc. 28th ACM Symposium on Principles of Programming Languages, ACM Press (2001) 104–115
12. Blanchet, B.: An efficient cryptographic protocol verifier based on prolog rules. In: Proc. 14th IEEE Computer Security Foundations Workshop, IEEE CS (2001) 82–96
13. Delaune, S., Kremer, S., Ryan, M.D.: Verifying privacy-type properties of electronic voting protocols. Journal of Computer Security **17**(4) (2009) 435–487
14. Jonker, H.L., Mauw, S., Pang, J.: A formal framework for quantifying voter-controlled privacy. Journal of Algorithms in Cognition, Informatics and Logic **64**(2-3) (2009) 89–105
15. Dong, N., Jonker, H.L., Pang, J.: Analysis of a receipt-free auction protocol in the applied pi calculus. In: Proc. 7th Workshop on Formal Aspects in Security and Trust. Volume 6561 of LNCS., Springer (2011) 223–238
16. Schneider, S., Sidiropoulos, A.: CSP and anonymity. In: Proc. 4th European Symposium on Research in Computer Security. Volume 1146 of LNCS., Springer (1996) 198–218
17. van Deursen, T., Mauw, S., Radomirović, S.: Untraceability of RFID protocols. In: Proc. 2nd Workshop on Information Security Theory and Practices. Smart Devices, Convergence and Next Generation. Volume 5019 of LNCS., Springer (2008) 1–15
18. Backes, M., Hriţcu, C., Maffei, M.: Automated verification of remote electronic voting protocols in the applied pi-calculus. In: Proc. 21st IEEE Computer Security Foundations Symposium, IEEE CS (2008) 195–209
19. Küsters, R., Truderung, T.: An epistemic approach to coercion-resistance for electronic voting protocols. In: Proc. 30th IEEE Symposium on Security and Privacy, IEEE CS (2009) 251–266
20. Arapinis, M., Chothia, T., Ritter, E., Ryan, M.: Analysing unlinkability and anonymity using the applied pi calculus. In: CSF. (2010) 107–121

21. Küsters, R., Truderung, T., Vogt, A.: A game-based definition of coercion-resistance and its applications. In: Proc. 23rd IEEE Computer Security Foundations Symposium, IEEE CS (2010) 122–136
22. Brands, S.A.: Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press (2000)
23. Backes, M., Maffei, M., Unruh, D.: Zero-knowledge in the applied pi-calculus and automated verification of the direct anonymous attestation protocol. In: Proc. IEEE Symposium on Security and Privacy. (2008) 202–215
24. Blanchet, B.: From secrecy to authenticity in security protocols. In: Proc. Static Analysis, 9th International Symposium. (2002) 342–359

A Functions and equational theory

fun	true/0.
fun	hash/3.
fun	pk/1.
fun	enc/2.
fun	commit/2.
fun	sign(/, *2*).
private fun	drcred/2.
private fun	ptcred/5.
fun	zk/2.
fun	spk/3.
fun	invoice/1.
fun	key/1.
fun	host/1.

Fig. 20. Functions.

$$
\begin{array}{ll}
\textit{reduc} & \mathsf{dec}(\mathsf{enc}(m, \mathsf{pk}(sk)), sk) = m. \\
\textit{reduc} & \mathsf{open}(\mathsf{commit}(x, y), y) = x. \\
\textit{reduc} & \mathsf{Vfy\text{-}sign}(\mathsf{sign}(x, y), \mathsf{pk}(y)) = \mathsf{true}. \\
\textit{reduc} & \mathsf{getsignmsg}(\mathsf{sign}(x, y), \mathsf{pk}(y)) = x. \\
\textit{reduc} & \mathsf{getpublic}(\mathsf{zk}(x, y)) = y. \\
\textit{reduc} & \mathsf{getSpkMsg}(\mathsf{spk}(x, y, z)) = z. \\
\textit{reduc} & \mathsf{getSpkVinfo}(\mathsf{spk}(x, y, z)) = y. \\
\mathtt{equation} & \mathsf{key}(\mathsf{host}(x)) = x. \\
\mathtt{equation} & \mathsf{host}(\mathsf{key}(x)) = x. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{dr}}(\mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})), \\
 & \qquad\qquad \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad\quad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})), \\
 & \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad\quad (\mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}), \\
 & \qquad\qquad\quad\; \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}))), \\
 & \qquad\qquad \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}), \\
 & \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc})) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}), \\
 & \qquad\qquad\quad (\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})), \\
 & \qquad\qquad\quad (\mathtt{presc}, \mathit{PrescriptID}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \\
 & \qquad\qquad\quad\; \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}))), \\
 & \qquad\qquad \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathtt{presc}, \mathit{PrescriptID}, \\
 & \qquad\qquad \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \\
 & \qquad\qquad \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{PtAuthSss}}(\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad\quad (\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \mathtt{Sss}), \\
 & \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \mathtt{Sss}) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}spk}_{\mathsf{PtSpk}}(\mathsf{spk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}, r_{pt}), \\
 & \qquad\qquad\quad (\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})), \\
 & \qquad\qquad\quad \mathtt{nonce}), \\
 & \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}), \mathtt{nonce}) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad\quad (\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad\quad \mathsf{enc}(\mathtt{Hii}, pk_x))), \\
 & \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad \mathsf{enc}(\mathtt{Hii}, pk_x), pk_x) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(\mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \\
 & \qquad\qquad\quad (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}), \\
 & \qquad\qquad\qquad (\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})), \\
 & \qquad\qquad\qquad (\mathtt{presc}, \mathit{PrescriptID}, \\
 & \qquad\qquad\qquad\; \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c_{ph}_\mathit{Comt}_{pt})), \\
 & \qquad\qquad\quad \mathsf{enc}(\mathtt{Pnym}_{dr}, pk_x))), \\
 & \qquad\qquad \mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}), \\
 & \qquad\qquad\quad (\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})), \\
 & \qquad\qquad\quad (\mathtt{presc}, \mathit{PrescriptID}, \\
 & \qquad\qquad\quad\; \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c_{ph}_\mathit{Comt}_{pt})), \\
 & \qquad\qquad \mathsf{enc}(\mathtt{Pnym}_{dr}, pk_x), pk_x) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(\mathsf{zk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad\quad (\mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \mathsf{enc}(\mathtt{Pnym}_{pt}, pk_x))), \\
 & \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad \mathsf{enc}(\mathtt{Pnym}_{pt}, pk_x), pk_x) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(\mathsf{spk}((\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad\quad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad\quad (c_\mathit{PrescriptID}, c_{pt}_\mathit{Id}_{ph}, vc_1, vc_2, vc_3, vc_3', vc_4, c_5)), \\
 & \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt}, \mathtt{Pnym}_{pt}, \mathtt{Hii}, \mathtt{Sss}, \mathtt{Acc}), \\
 & \qquad\qquad c_\mathit{PrescriptID}, c_{pt}_\mathit{Id}_{ph}, vc_1, vc_2, vc_3, vc_3', vc_4, c_5) = \mathsf{true}.
\end{array}
$$

Fig. 21. Equational theory.

B Modelling of the protocol with revised assumptions

According to the revised assumptions **s2’**, **s3’**, **s4’**, **s4”**, **s5’** and **s6’**, we revise the following parts of the model. Figure 22 shows the parts differing from Figure 20. Figure 23 shows the parts differing from Figure 21.

```
fun            penc/3.
private  fun   drcred/3.
private  fun   ptcred/6.
```

Fig. 22. Functions with revised assumptions.

Roles R_{ph}, R_{mpa} and R_{hii} are the same as in Figure 16, Figure 17 and Figure 18, respectively.

Processes $P_{ph}\_p_1$, $P_{ph}\_p_2$, $P_{mpa}\_p_1$, $P_{mpa}\_p_2$, and P_{hii} are the same as in Figure reffig:proph1, Figure reffig:proph2, Figure reffig:prompa1, Figure reffig:prompa2 and Figure reffig:prohii, respectively.

$$
\begin{array}{ll}
\textit{reduc} & \mathsf{pdec}(\mathsf{penc}(\mathsf{m},\mathsf{pk}(\mathsf{sk}),\mathsf{r}),\mathsf{sk}) = \mathsf{m}.\\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{dr}}(\mathsf{zk}((\mathtt{Pnym}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr}),\mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr})),\\
& \qquad\qquad \mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr})) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(\mathsf{zk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}))),\\
& \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}))) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(\mathsf{zk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt})),\\
& \qquad\qquad\qquad (\mathsf{commit}(\mathtt{Id}_{pt},r_{pt}),\\
& \qquad\qquad\qquad\; \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt})))),\\
& \qquad\qquad \mathsf{commit}(\mathtt{Id}_{pt},r_{pt}),\\
& \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}))) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(\mathsf{spk}((\mathtt{Pnym}_{dr},\mathtt{r}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr}),\\
& \qquad\qquad\qquad\quad (\mathsf{commit}(\mathtt{Pnym}_{dr},\mathtt{r}_{dr}),\mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr})),\\
& \qquad\qquad\qquad\quad (\mathtt{presc},\mathit{PrescriptID},\mathsf{commit}(\mathtt{Pnym}_{dr},\mathtt{r}_{dr}),\\
& \qquad\qquad\qquad\quad\; \mathsf{commit}(\mathtt{Id}_{pt},r_{pt}))),\\
& \qquad\qquad\quad \mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr}),\mathtt{presc},\mathit{PrescriptID},\\
& \qquad\qquad\quad \mathsf{commit}(\mathtt{Pnym}_{dr},\mathtt{r}_{dr}),\\
& \qquad\qquad\quad \mathsf{commit}(\mathtt{Id}_{pt},r_{pt})) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{PtAuthSss}}(\mathsf{zk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad\qquad (\mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\mathtt{Sss}),\\
& \qquad\qquad\quad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\mathtt{Sss}) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}spk}_{\mathsf{PtSpk}}(\mathsf{spk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},r_{pt},\mathtt{n}_{pt}),\\
& \qquad\qquad\qquad (\mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\mathsf{commit}(\mathtt{Id}_{pt},r_{pt})),\\
& \qquad\qquad\qquad \mathtt{nonce}),\\
& \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad \mathsf{commit}(\mathtt{Id}_{pt},r_{pt}),\mathtt{nonce}) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(\mathsf{zk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt},\mathtt{r}'),\\
& \qquad\qquad\qquad (\mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad\qquad\; \mathsf{penc}(\mathtt{Hii},pk_x,\mathtt{r}'))),\\
& \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad \mathsf{penc}(\mathtt{Hii},pk_x,\mathtt{r}'),pk_x) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(\mathsf{zk}((\mathtt{Pnym}_{dr},\mathtt{r}_{dr},\mathtt{r}'),\\
& \qquad\qquad\qquad\quad (\mathsf{spk}((\mathtt{Pnym}_{dr},\mathtt{r}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr}),\\
& \qquad\qquad\qquad\qquad\quad (\mathsf{commit}(\mathtt{Pnym}_{dr},\mathtt{r}_{dr}),\mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr})),\\
& \qquad\qquad\qquad\qquad\quad (\mathtt{presc},\mathit{PrescriptID},\\
& \qquad\qquad\qquad\qquad\quad\; \mathsf{commit}(\mathtt{Pnym}_{dr},\mathtt{r}_{dr}),c_{ph}_Comt_{pt})),\\
& \qquad\qquad\qquad\quad\; \mathsf{penc}(\mathtt{Pnym}_{dr},pk_x,\mathtt{r}'))),\\
& \qquad\qquad\quad \mathsf{spk}((\mathtt{Pnym}_{dr},\mathtt{r}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr}),\\
& \qquad\qquad\qquad\quad (\mathsf{commit}(\mathtt{Pnym}_{dr},\mathtt{r}_{dr}),\mathsf{drcred}(\mathtt{Pnym}_{dr},\mathtt{Id}_{dr},\mathtt{n}_{dr})),\\
& \qquad\qquad\qquad\quad (\mathtt{presc},\mathit{PrescriptID},\\
& \qquad\qquad\qquad\quad\; \mathsf{commit}(\mathtt{Pnym}_{dr},\mathtt{r}_{dr}),c_{ph}_Comt_{pt})),\\
& \qquad\qquad\quad \mathsf{penc}(\mathtt{Pnym}_{dr},pk_x,\mathtt{r}'),pk_x) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(\mathsf{zk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt},\mathtt{r}'),\\
& \qquad\qquad\qquad (\mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\mathsf{penc}(\mathtt{Pnym}_{pt},pk_x,\mathtt{r}'))),\\
& \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad \mathsf{penc}(\mathtt{Pnym}_{pt},pk_x,\mathtt{r}'),pk_x) = \mathsf{true}.\\
\textit{reduc} & \mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(\mathsf{spk}((\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad\qquad\quad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad\qquad\quad (c_PrescriptID,c_{pt}_Id_{ph},vc_1,vc_2,vc_3,vc_3',vc_4,c_5)),\\
& \qquad\qquad \mathsf{ptcred}(\mathtt{Id}_{pt},\mathtt{Pnym}_{pt},\mathtt{Hii},\mathtt{Sss},\mathtt{Acc},\mathtt{n}_{pt}),\\
& \qquad\qquad c_PrescriptID,c_{pt}_Id_{ph},vc_1,vc_2,vc_3,vc_3',vc_4,c_5) = \mathsf{true}.
\end{array}
$$

Fig. 23. Equational theory with revised assumptions.

```
$DLV =$
        $\nu\mathtt{sk}_{sso}; \nu\mathtt{ch}_{hp}; \nu\mathtt{ch}_{mp}; \nu\mathtt{ch}_{phpt};$
        let $\mathtt{pk}_{sso} = \mathsf{pk}(\mathtt{sk}_{sso})$ in
        $\mathsf{out}(\mathtt{ch}, \mathtt{pk}_{sso});$
        $(!(R_{dr})\ |!(R_{ph})\ |!(R_{mpa})\ |!(R_{hii})\ |$
        $(\mathsf{in}(\mathtt{ch}_{hp}, \mathtt{Hii});$ let $c_{pt}\_pk_{hii} = \mathsf{key}(\mathtt{Hii})$ in $!R_{pt}))$
```

Fig. 24. The DLVV08 protocol with revised assumptions.

```
$R_{dr} := \nu\mathtt{Id}_{dr};$
        $!(\nu\mathtt{Pnym}_{dr}; P_{dr})$
```

Fig. 25. Process R_{dr} with revised assumptions.

```
$R_{pt} := \nu\mathtt{Id}_{pt};$
       $\nu\mathtt{Pnym}_{pt}; \nu\mathtt{Acc};$                        } $init_{pt}$
       $!(\nu\mathtt{Sss}; \mathsf{in}(\mathtt{ch}_{phpt}, rcv\_pk_{ph});$                                  } $P_{pt}$
        let $rcv_{pt}\_pk_{ph} = rcv\_pk_{ph}$ in let $\mathtt{Id}_{ph} = \mathsf{host}(rcv\_pk_{ph})$ in
        $(\ldots$                                                          } $P_{pt}\_p_1$
        let $c\_Pnym_{dr} = \mathsf{open}(c\_Comt_{dr}, rcv\_r_{dr})$ in
        $\mathsf{in}(\mathtt{ch}, rcv\_Auth_{ph});$                                 } $P_{pt}\_p_2$
        $\ldots)$
```

Fig. 26. Process R_{pt} with revised assumptions.

```
let $P_{dr} =$
            $\nu\mathtt{n}_{dr};$
            $\mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})));$
            $\mathsf{in}(\mathtt{ch}, (rcv\_Auth_{pt}, rcv\_PtProof));$
            let $c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt})$ in
            let $(c\_Comt_{pt}, = c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)$ in
            if $\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt}) = \mathsf{true}$ then
            if $\mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt})) = \mathsf{true}$ then
            $\nu\mathtt{presc};$
            $\nu\mathtt{r}_{dr};$
            let $PrescriptID = \mathsf{hash}(\mathtt{presc}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))$ in
            $\mathsf{out}(\mathtt{ch}_{dp}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}),$
                        $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})),$
                        $(\mathtt{presc}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
                  $\mathtt{r}_{dr})).$
```

Fig. 27. The doctor process P_{dr} with revised assumptions.

```
let $P_{pt}$ =
        in(ch, $rcv\_Auth_{dr}$);
        let $c\_Cred_{dr}$ = getpublic($rcv\_Auth_{dr}$) in
        if Vfy-zk$_{\mathsf{Auth}_{dr}}$($rcv\_Auth_{dr}$, $c\_Cred_{dr}$) = true then
        $\nu r_{pt}$;
        $\nu \mathtt{n}_{pt}$;
        out(ch, (zk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                    ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$)),
                 zk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                    (commit($\mathtt{Id}_{pt}$, $r_{pt}$),
                     ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$)))));
        in($\mathsf{ch}_{dp}$, ($rcv\_PrescProof$, $rcv\_r_{dr}$));
        let ($c\_presc$, $c\_PrescriptID$, $c\_Comt_{dr}$, = commit($\mathtt{Id}_{pt}$, $r_{pt}$))
           = getSpkMsg($rcv\_PrescProof$) in
        if Vfy-spk$_{\mathsf{PrescProof}}$($rcv\_PrescProof$, ($c\_Cred_{dr}$, $c\_presc$, $c\_PrescriptID$,
                              $c\_Comt_{dr}$, commit($\mathtt{Id}_{pt}$, $r_{pt}$))) = true then
        let $c\_Pnym_{dr}$ = open($c\_Comt_{dr}$, $rcv\_r_{dr}$) in
        in(ch, $rcv\_Auth_{ph}$);
        if Vfy-sign($rcv\_Auth_{ph}$, $rcv_{pt}\_pk_{ph}$) = true then
        let (= $c_{pt}\_Id_{ph}$, $c_{pt}\_Id_{mpa}$) = getsignmsg($rcv\_Auth_{ph}$, $rcv_{pt}\_pk_{ph}$) in
        let $c_{pt}\_pk_{mpa}$ = key($c_{pt}\_Id_{mpa}$) in
        out(ch, zk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                   (ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$), Sss)));
        $\nu$nonce;
        $\nu \mathtt{r}'$;
        let $vc_1$ = zk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$, $\mathtt{r}'$),
                     (ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                      penc(Hii, $c_{pt}\_pk_{mpa}$, $\mathtt{r}'$))) in
        let $vc_2$ = zk(($c\_Pnym_{dr}$, $rcv\_r_{dr}$, $\mathtt{r}'$), ($rcv\_PrescProof$,
                      penc($c\_Pnym_{dr}$, $c_{pt}\_pk_{mpa}$, $\mathtt{r}'$))) in
        let $vc_3$ = zk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$, $\mathtt{r}'$),
                     (ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                      penc($\mathtt{Pnym}_{pt}$, $\mathtt{pk}_{sso}$, $\mathtt{r}'$))) in
        let $vc_3'$ = zk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$, $\mathtt{r}'$),
                     (ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                      penc(Hii, $\mathtt{pk}_{sso}$, $\mathtt{r}'$))) in
        let $vc_4$ = zk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$, $\mathtt{r}'$),
                     (ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                      penc($\mathtt{Pnym}_{pt}$, $c_{pt}\_pk_{mpa}$, $\mathtt{r}'$))) in
        let $vc_5$ = zk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$, $\mathtt{r}'$),
                     (ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                      penc($\mathtt{Pnym}_{pt}$, $c_{pt}\_pk_{hii}$, $\mathtt{r}'$))) in
        let $c_5$ = penc($vc_5$, $c_{pt}\_pk_{mpa}$, $\mathtt{r}'$) in
        out($\mathsf{ch}_{ptph}$, ($rcv\_PrescProof$,
                 spk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
                     (ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$), commit($\mathtt{Id}_{pt}$, $r_{pt}$)),
                     nonce),
                 $vc_1$, $vc_2$, $vc_3$, $vc_3'$, $vc_4$, $c_5$));
        in($\mathsf{ch}_{phpt}$, $rcv\_Invoice$);
        let $ReceiptAck$ = spk(($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
        ptcred($\mathtt{Id}_{pt}$, $\mathtt{Pnym}_{pt}$, Hii, Sss, Acc, $\mathtt{n}_{pt}$),
        ($c\_PrescriptID$, $c_{pt}\_Id_{ph}$, $vc_1$, $vc_2$, $vc_3$, $vc_3'$, $vc_4$, $c_5$)) in
        out($\mathsf{ch}_{ptph}$, $ReceiptAck$).
```

Fig. 28. The patient process P_{pt} with revised assumptions.

C Fixing enforced prescribing-privacy with untappable channels

C.1 Based on protocol with assumption s4'

The following model is based on the assumptions **s4'** and **s8'**. Other assumptions follow the original ones. We only show the parts need to be revised. The parts of the model not mentioned in this section follow the original model.

Functions and equational theory which are different from in Figure 20 or Figure fig:eqtheory are shown in Figure 29. Roles R_{dr}, R_{pt}, R_{ph}, R_{mpa} and R_{hii} are the same as in Figure 14, Figure 15, Figure 16, Figure 17 and Figure 18, respectively.

$$
\begin{array}{ll}
\textit{private fun} & \mathsf{drcred}/3. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{dr}}(\mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})), \\
 & \qquad \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \\
 & \qquad\qquad (\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})), \\
 & \qquad\qquad (\mathtt{presc}, \mathit{PrescriptID}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \\
 & \qquad\qquad\; \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt}))), \\
 & \qquad \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \mathtt{presc}, \mathit{PrescriptID}, \\
 & \qquad \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \\
 & \qquad \mathsf{commit}(\mathtt{Id}_{pt}, r_{pt})) = \mathsf{true}. \\
\textit{reduc} & \mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(\mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr},), \\
 & \qquad\qquad (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \\
 & \qquad\qquad\qquad (\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})), \\
 & \qquad\qquad\qquad (\mathtt{presc}, \mathit{PrescriptID}, \\
 & \qquad\qquad\qquad\; \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c_{ph}\_\mathit{Comt}_{pt})), \\
 & \qquad\qquad\; \mathsf{enc}(\mathtt{Pnym}_{dr}, pk_x))), \\
 & \qquad \mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \\
 & \qquad\qquad (\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})), \\
 & \qquad\qquad (\mathtt{presc}, \mathit{PrescriptID}, \\
 & \qquad\qquad\; \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c_{ph}\_\mathit{Comt}_{pt})), \\
 & \qquad \mathsf{enc}(\mathtt{Pnym}_{dr}, pk_x), pk_x) = \mathsf{true}.
\end{array}
$$

Fig. 29. Functions and equational theory (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

```
$DLV =$
        $\nu \mathtt{sk}_{sso}; \nu \mathtt{ch}_{hp}; \nu \mathtt{ch}_{mp}; \nu \mathtt{ch}_{phpt};$
        $\nu \mathtt{ch}_{dp}; \nu \mathtt{ch}_{ptph}; \nu \mathtt{ch}_{pm}; \nu \mathtt{ch}_{hm}; \nu \mathtt{ch'}_{phpt}; \nu \mathtt{ch}_{mh};$
        $\mathtt{let}\ \mathtt{pk}_{sso} = \mathsf{pk}(\mathtt{sk}_{sso})\ \mathtt{in}$
        $\mathsf{out}(\mathtt{ch}, \mathtt{pk}_{sso});$
        $(!(R_{dr})\ |!(R_{pt})\ |!(R_{ph})\ |!(R_{mpa})\ |!(R_{hii}))$
```

Fig. 30. The DLVV08 protocol (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

```
$\mathtt{let}\ P_{dr} =$
          $\nu \mathtt{n}_{dr};$
          $\mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})));$
          $\mathsf{in}(\mathtt{ch}, (rcv\_Auth_{pt}, rcv\_PtProof));$
          $\mathtt{let}\ c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt})\ \mathtt{in}$
          $\mathtt{let}\ (c\_Comt_{pt}, = c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)\ \mathtt{in}$
          $\mathtt{if}\ \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt}) = \mathsf{true}\ \mathtt{then}$
          $\mathtt{if}\ \mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt})) = \mathsf{true}\ \mathtt{then}$
          $\nu \mathtt{presc};$
          $\nu \mathtt{r}_{dr};$
          $\mathtt{let}\ PrescriptID = \mathsf{hash}(\mathtt{presc}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))\ \mathtt{in}$
          $\mathsf{out}(\mathtt{ch}_{dp}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}),$
                    $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})),$
                    $(\mathtt{presc}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
               $\mathtt{r}_{dr})).$
```

Fig. 31. The doctor process P_{dr} (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

```
let $P_{pt}$ =
        in(ch, $rcv\_Auth_{dr}$);
        let $c\_Cred_{dr}$ = getpublic($rcv\_Auth_{dr}$) in
        if Vfy-zk$_{\mathsf{Auth}_{dr}}$($rcv\_Auth_{dr}, c\_Cred_{dr}$) = true then
        $\nu r_{pt}$;
        out(ch, (zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                  ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc)),
                zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                  (commit(Id$_{pt}$, $r_{pt}$),
                   ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc)))));
        in(ch$_{dp}$, ($rcv\_PrescProof, rcv\_r_{dr}$));
        let ($c\_presc, c\_PrescriptID, c\_Comt_{dr}$, = commit(Id$_{pt}$, $r_{pt}$))
           = getSpkMsg($rcv\_PrescProof$) in
        if Vfy-spk$_{\mathsf{PrescProof}}$($rcv\_PrescProof$, ($c\_Cred_{dr}, c\_presc, c\_PrescriptID$,
                          $c\_Comt_{dr}$, commit(Id$_{pt}$, $r_{pt}$))) = true then
        let $c\_Pnym_{dr}$ = open($c\_Comt_{dr}, rcv\_r_{dr}$) in
        in(ch, $rcv\_Auth_{ph}$);
        if Vfy-sign($rcv\_Auth_{ph}, rcv_{pt}\_pk_{ph}$) = true then
        let (= $c_{pt}\_Id_{ph}, c_{pt}\_Id_{mpa}$) = getsignmsg($rcv\_Auth_{ph}, rcv_{pt}\_pk_{ph}$) in
        let $c_{pt}\_pk_{mpa}$ = key($c_{pt}\_Id_{mpa}$) in
        out(ch, zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                 (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc), Sss)));
        $\nu$nonce;
        let $vc_1$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Hii, $c_{pt}\_pk_{mpa}$))) in
        let $vc_2$ = zk(($c\_Pnym_{dr}, rcv\_r_{dr}$), ($rcv\_PrescProof$,
                     enc($c\_Pnym_{dr}, c_{pt}\_pk_{mpa}$))) in
        let $vc_3$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Pnym$_{pt}$, pk$_{sso}$))) in
        let $vc'_3$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Hii, pk$_{sso}$))) in
        let $vc_4$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Pnym$_{pt}$, $c_{pt}\_pk_{mpa}$))) in
        let $vc_5$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Pnym$_{pt}$, $c_{pt}\_pk_{hii}$))) in
        let $c_5$ = enc($vc_5, c_{pt}\_pk_{mpa}$) in
        out(ch$_{ptph}$, ($rcv\_PrescProof$,
                spk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                   (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc), commit(Id$_{pt}$, $r_{pt}$)),
                   nonce),
                $vc_1, vc_2, vc_3, vc'_3, vc_4, c_5$));
        in(ch$'_{phpt}$, $rcv\_Invoice$);
        let $ReceiptAck$ = spk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
        ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
        ($c\_PrescriptID, c_{pt}\_Id_{ph}, vc_1, vc_2, vc_3, vc'_3, vc_4, c_5$)) in
        out(ch$_{ptph}$, $ReceiptAck$).
```

Fig. 32. The patient process P_{pt} (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

```
let $P_{ph}$ =
        $\mathsf{out}(\mathsf{ch}, \mathsf{sign}((\mathtt{Id}_{ph}, c_{ph}\_Id_{mpa}), \mathtt{sk}_{ph}))$;
        $\mathsf{in}(\mathsf{ch}, rcv\_PtAuthSss)$;
        let $(c_{ph}\_Cred_{pt}, c_{ph}\_Sss) = \mathsf{getpublic}(rcv\_PtAuthSss)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{PtAuthSss}}(rcv\_PtAuthSss, (c_{ph}\_Cred_{pt}, c_{ph}\_Sss)) = \mathsf{true}$ then
        $\mathsf{in}(\mathsf{ch}_{ptph}, (rcv_{ph}\_PrescProof, rcv_{ph}\_PtSpk,$
                $rcv\_vc_1, rcv\_vc_2, rcv\_vc_3, rcv\_vc_3', rcv\_vc_4, rcv\_c_5))$;
        let $(c_{ph}\_Comt_{dr}, c_{ph}\_Cred_{dr}) = \mathsf{getSpkVinfo}(rcv_{ph}\_PrescProof)$ in
        let $(c_{ph}\_presc, c_{ph}\_PrescriptID, = c_{ph}\_Comt_{dr}, c_{ph}\_Comt_{pt})$
            $= \mathsf{getSpkMsg}(rcv_{ph}\_PrescProof)$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(rcv_{ph}\_PrescProof, (c_{ph}\_Cred_{dr}, c_{ph}\_presc, c_{ph}\_PrescriptID,$
                        $c_{ph}\_Comt_{dr}, c_{ph}\_Comt_{pt})) = \mathsf{true}$ then
        let $c\_msg = \mathsf{getSpkMsg}(rcv_{ph}\_PtSpk)$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{PtSpk}}(rcv_{ph}\_PtSpk,$
                        $(c_{ph}\_Cred_{pt}, c_{ph}\_Comt_{pt}, c\_msg)) = \mathsf{true}$ then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_1) = \mathsf{getpublic}(rcv\_vc_1)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv\_vc_1, (c_{ph}\_Cred_{pt}, c\_Enc_1, rcv_{ph}\_pk_{mpa})) = \mathsf{true}$ then
        let $(= rcv_{ph}\_PrescProof, c\_Enc_2) = \mathsf{getpublic}(rcv\_vc_2)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(rcv\_vc_2, (rcv_{ph}\_PrescProof,$
                                        $c\_Enc_2, rcv_{ph}\_pk_{mpa})) = \mathsf{true}$ then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_3) = \mathsf{getpublic}(rcv\_vc_3)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv\_vc_3, (c_{ph}\_Cred_{pt}, c\_Enc_3, \mathsf{pk}_{sso})) = \mathsf{true}$ then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_3') = \mathsf{getpublic}(rcv\_vc_3')$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv\_vc_3', (c_{ph}\_Cred_{pt}, c\_Enc_3', \mathsf{pk}_{sso})) = \mathsf{true}$ then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_4) = \mathsf{getpublic}(rcv\_vc_4)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv\_vc_4,$
                        $(c_{ph}\_Cred_{pt}, c\_Enc_4, rcv_{ph}\_pk_{mpa})) = \mathsf{true}$ then
        $\mathsf{out}(\mathsf{ch'}_{phpt}, \mathsf{invoice}(c_{ph}\_PrescriptID))$;
        $\mathsf{in}(\mathsf{ch}_{ptph}, rcv\_ReceiptAck)$;
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv\_ReceiptAck, (c_{ph}\_Cred_{pt}, c_{ph}\_PrescriptID,$
        $\mathtt{Id}_{ph}, rcv\_vc_1, rcv\_vc_2, rcv\_vc_3, rcv\_vc_3', rcv\_vc_4, rcv\_c_5)) = \mathsf{true}$ then
        $\mathsf{out}(\mathsf{ch}, (\mathsf{sign}((\mathtt{Id}_{ph}, c_{ph}\_Id_{mpa}), \mathtt{sk}_{ph}), \mathtt{Id}_{ph}))$;
        $\mathsf{in}(\mathsf{ch}, rcv\_Auth_{mpa})$;
        if $\mathsf{Vfy\text{-}sign}(rcv\_Auth_{mpa}, rcv_{ph}\_pk_{mpa}) = \mathsf{true}$ then
        $\mathsf{out}(\mathsf{ch}_{pm}, (rcv_{ph}\_PrescProof,$
                $rcv\_vc_1, rcv\_vc_2, rcv\_vc_3, rcv\_vc_3', rcv\_vc_4, rcv\_c_5,$
                $rcv\_ReceiptAck))$
```

Fig. 33. The pharmacist process P_{ph} (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

```
let $P_{mpa}$ =
        in(ch, $(rcv_{mpa}\_Auth_{ph}, c_{mpa}\_Id_{ph})$);
        let $rcv_{mpa}\_pk_{ph}$ = key($c_{mpa}\_Id_{ph}$) in
        Vfy-sign($rcv_{mpa}\_Auth_{ph}, rcv_{mpa}\_pk_{ph}$) = true
        let (= $c_{mpa}\_Id_{ph}$, = Id$_{mpa}$)
            = getSpkMsg($rcv_{mpa}\_Auth_{ph}, rcv_{mpa}\_pk_{ph}$) in
        out(ch, sign(Id$_{mpa}$, sk$_{mpa}$));
        in(ch$_{pm}$, $(rcv_{mpa}\_PrescProof, rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2, rcv_{mpa}\_vc_3,$
                $rcv_{mpa}\_vc'_3, rcv_{mpa}\_vc_4, rcv_{mpa}\_c_5, rcv_{mpa}\_ReceiptAck)$);
        let $(c_{mpa}\_Comt_{dr}, c_{mpa}\_Cred_{dr})$ = getSpkVinfo($rcv_{mpa}\_PrescProof$) in
        let $(c_{mpa}\_presc, c_{mpa}\_PrescriptID,$ = $c_{mpa}\_Comt_{dr}, c_{mpa}\_Comt_{pt})$
            = getSpkMsg($rcv_{mpa}\_PrescProof$) in
        if Vfy-spk$_{\mathsf{PrescProof}}$($rcv_{mpa}\_PrescProof, (c_{mpa}\_Cred_{dr}, c_{mpa}\_presc,$
                  $c_{mpa}\_PrescriptID, c_{mpa}\_Comt_{dr}, c_{mpa}\_Comt_{pt})$) = true then
        let (= $c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_1$) = getpublic($rcv_{mpa}\_vc_1$) in
        if Vfy-zk$_{\mathsf{VEncHii}}$($rcv_{mpa}\_vc_1$,
                        $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_1$, pk$_{mpa}$)) = true then
        let $c_{mpa}\_Hii$ = dec($c_{mpa}\_Enc_1$, sk$_{mpa}$) in
        let (= $rcv_{mpa}\_PrescProof, c_{mpa}\_Enc_2$) = getpublic($rcv_{mpa}\_vc_2$) in
        if Vfy-zk$_{\mathsf{VEncDrnymMpa}}$($rcv_{mpa}\_vc_2$,
                          ($rcv_{mpa}\_PrescProof, c_{mpa}\_Enc_2$, pk$_{mpa}$)) = true then
        let $c_{mpa}\_Pnym_{dr}$ = dec($c_{mpa}\_Enc_2$, sk$_{mpa}$) in
        let (= $c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc3$) = getpublic($rcv_{mpa}\_vc_3$) in
        if Vfy-zk$_{\mathsf{VEncPtnym}}$($rcv_{mpa}\_vc_3$,
                          ($c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_3$, pk$_{sso}$)) = true then
        let (= $c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc'_3$) = getpublic($rcv_{mpa}\_vc'_3$) in
        if Vfy-zk$_{\mathsf{VEncHii}}$($rcv_{mpa}\_vc'_3$,
                        ($c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc'_3$, pk$_{sso}$)) = true then
        let (= $c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_4$) = getpublic($rcv_{mpa}\_vc_4$) in
        if Vfy-zk$_{\mathsf{VEncPtnym}}$($rcv_{mpa}\_vc_4$,
                          ($c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_4$, pk$_{mpa}$)) = true then
        let $c_{mpa}\_Pnym_{pt}$ = dec($c_{mpa}\_Enc_4$, sk$_{mpa}$) in
        if Vfy-spk$_{\mathsf{ReceiptAck}}$($rcv_{mpa}\_ReceiptAck, (c_{mpa}\_Cred_{pt},$
            $c_{mpa}\_PrescriptID, c_{mpa}\_Id_{ph}, rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2, rcv_{mpa}\_vc_3,$
            $rcv_{mpa}\_vc'_3, rcv_{mpa}\_vc_4, rcv_{mpa}\_c_5)$)
          = true then
        out(ch, (sign(Id$_{mpa}$, sk$_{mpa}$), Id$_{mpa}$));
        in(ch, $rcv_{mpa}\_Auth_{hii}$);
        let $c_{mpa}\_pk_{hii}$ = key($c_{mpa}\_Hii$) in
        if Vfy-sign($rcv_{mpa}\_Auth_{hii}, c_{mpa}\_pk_{hii}$) = true then
        if getsignmsg($rcv_{mpa}\_Auth_{hii}, c_{mpa}\_pk_{hii}$) = $c_{mpa}\_Hii$ then
        out(ch$_{mh}$, ($rcv_{mpa}\_ReceiptAck$, dec($rcv_{mpa}\_c_5$, sk$_{mpa}$)));
        in(ch$_{hm}$, $rcv_{mpa}\_Invoice$).
```

Fig. 34. The MPA process P_{mpa} (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

```
let $P_{hii}$ =
          $\mathsf{in}(\mathsf{ch}, (rcv_{hii}\_Auth_{mpa}, rcv_{hii}\_Id_{mpa}))$;
          let $c_{hii}\_pk_{mpa} = \mathsf{key}(rcv_{hii}\_Id_{mpa})$ in
          if $\mathsf{Vfy\text{-}sign}(rcv_{hii}\_Auth_{mpa}, c_{hii}\_pk_{mpa}) =$ true then
          $\mathsf{out}(\mathsf{ch}, \mathsf{sign}(\mathtt{Id}_{hii}, \mathtt{sk}_{hii}))$;
          $\mathsf{in}(\mathsf{ch}_{mh}, (rcv_{hii}\_ReceiptAck, c_{hii}\_vc_5))$;
          let $c_{hii}\_Cred_{pt} = \mathsf{getSpkVinfo}(rcv_{hii}\_ReceiptAck)$ in
          let $(c_{hii}\_PrescriptID, c_{hii}\_Id_{ph}, c_{hii}\_vc_1, c_{hii}\_vc_2, c_{hii}\_vc_3, c_{hii}\_vc_3', c_{hii}\_vc_4,$
                $c_{hii}\_c_5) = \mathsf{getSpkMsg}(rcv_{hii}\_ReceiptAck)$ in
          if $\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv_{hii}\_ReceiptAck, (c_{hii}\_Cred_{pt},$
              $c_{hii}\_PrescriptID, c_{hii}\_Id_{ph}, c_{hii}\_vc_1, c_{hii}\_vc_2, c_{hii}\_vc_3, c_{hii}\_vc_3',$
              $c_{hii}\_vc_4, c_{hii}\_c_5)) =$ true then
          let $(= c_{hii}\_Cred_{pt}, c_{hii}\_Enc_5) = \mathsf{getpublic}(c_{hii}\_vc_5)$ in
          if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(c_{hii}\_vc_5, (c_{hii}\_Cred_{pt}, c_{hii}\_Enc_5, \mathtt{pk}_{hii})) = \mathsf{true}$
          then
          let $c_{hii}\_Pnym_{pt} = \mathsf{dec}(c_{hii}\_Enc_5, \mathtt{sk}_{hii})$ in
          $\mathsf{out}(\mathsf{ch}_{hm}, \mathsf{invoice}(c_{hii}\_PrescriptID))$.
```

Fig. 35. The HII process P_{hii} (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

C.2 Based on the original assumptions

The following models are based on the assumption **s8'**. Other assumptions follow the original ones. Most of the parts are the same as in the previous sub-section. We only list the different parts.

The functions and equational theory are the same as in the original model as in Figure 20 or Figure fig:eqtheory. The framework of the DLVV08 protocol is the same as in the previous sub-section as in Figure 30. Roles R_{dr}, R_{pt}, R_{ph}, R_{mpa} and R_{hii} are the same as in Figure 14, Figure 15, Figure 16, Figure 17 and Figure 18, respectively. Processes $P_{pt}\_p_1$, $P_{pt}\_p_2$, $P_{ph}\_p_1$, $P_{ph}\_p_2$, $P_{mpa}\_p_1$, $P_{mpa}\_p_2$, and P_{hii} are the same as in Figure 4, Figure 5, Figure reffig:proph1, Figure reffig:proph2, Figure reffig:prompa1, Figure reffig:prompa2 and Figure reffig:prohii, respectively.

```
$\mathit{init}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\}.(!P_{dr}\{\mathtt{A}/\mathit{Id}_{dr}\} \mid \mathit{main}_{dr}\{\mathtt{A}/\mathit{Id}_{dr}, \mathsf{a}/\mathit{presc}\}^{\mathsf{chc}}) =$
$\mathtt{let}\ \mathtt{Id}_{dr} = \mathtt{A}\ \mathtt{in}$
$\nu\mathtt{Pnym}_{dr};$
$(!P_{dr} \mid$
$(\mathsf{out}(\mathtt{chc}, \mathtt{Id}_{dr});$
$\mathsf{out}(\mathtt{chc}, \mathtt{Pnym}_{dr});$
$\nu\mathtt{n}_{dr};$
$\mathsf{out}(\mathtt{chc}, \mathtt{n}_{dr});$
$\mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})));$
$\mathsf{in}(\mathtt{ch}, (\mathit{rcv\_Auth}_{pt}, \mathit{rcv\_PtProof}));$
$\mathsf{out}(\mathtt{chc}, (\mathit{rcv\_Auth}_{pt}, \mathit{rcv\_PtProof}));$
$\mathtt{let}\ \mathit{c\_Cred}_{pt} = \mathsf{getpublic}(\mathit{rcv\_Auth}_{pt})\ \mathtt{in}$
$\mathtt{let}\ (\mathit{c\_Comt}_{pt}, = \mathit{c\_Cred}_{pt}) = \mathsf{getpublic}(\mathit{rcv\_PtProof})\ \mathtt{in}$
$\mathtt{if}\ \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(\mathit{rcv\_Auth}_{pt}, \mathit{c\_Cred}_{pt}) = \mathtt{true}\ \mathtt{then}$
$\mathtt{if}\ \mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(\mathit{rcv\_PtProof}, (\mathit{c\_Comt}_{pt}, \mathit{c\_Cred}_{pt})) = \mathtt{true}\ \mathtt{then}$
$\mathsf{out}(\mathtt{chc}, \mathsf{a});$
$\nu\mathtt{r}_{dr};$
$\mathsf{out}(\mathtt{chc}, \mathtt{r}_{dr});$
$\mathtt{let}\ \mathit{PrescriptID} = \mathsf{hash}(\mathsf{a}, \mathit{c\_Comt}_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))\ \mathtt{in}$
$\mathsf{out}(\mathtt{ch}_{dp}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}),$
                $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})),$
                $(\mathsf{a}, \mathit{PrescriptID}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathit{c\_Comt}_{pt})),$
        $\mathtt{r}_{dr}));$
$\mathsf{out}(\mathtt{chc}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}),$
                $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})),$
                $(\mathsf{a}, \mathit{PrescriptID}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathit{c\_Comt}_{pt})),$
        $\mathtt{r}_{dr})))).$
```

Fig. 36. The doctor process P_{dr}^{chc} (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

```
$init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid P'_{dr}\{\mathtt{A}/Id_{dr}\}) =$
let $\mathtt{Id}_{dr} = \mathtt{A}$ in
$\nu\mathtt{Pnym}_{dr}$;
$(!P_{dr} \mid$
$(\mathsf{out}(\mathtt{chc}, \mathtt{Id}_{dr})$;
$\mathsf{out}(\mathtt{chc}, \mathtt{Pnym}_{dr})$;
$\nu\mathtt{n}_{dr}$;
$\mathsf{out}(\mathtt{chc}, \mathtt{n}_{dr})$;
$\mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})))$;
$\mathsf{in}(\mathtt{ch}, (rcv\_Auth_{pt}, rcv\_PtProof))$;
$\mathsf{out}(\mathtt{chc}, (rcv\_Auth_{pt}, rcv\_PtProof))$;
let $c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt})$ in
let $(c\_Comt_{pt}, = c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)$ in
if $\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt}) = \mathsf{true}$ then
if $\mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt})) = \mathsf{true}$ then
$\mathsf{out}(\mathtt{chc}, \mathsf{a})$;
$\nu\mathtt{r}_{dr}$;
$\mathsf{out}(\mathtt{chc}, \mathtt{r}_{dr})$;
let $PrescriptID = \mathsf{hash}(\mathsf{b}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))$ in
$\mathsf{out}(\mathtt{ch}_{dp}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}),$
            $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})),$
            $(\mathsf{b}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
       $\mathtt{r}_{dr}))$;
$\mathsf{out}(\mathtt{chc}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr}),$
            $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}, \mathtt{n}_{dr})),$
            $(\mathsf{a}, \mathsf{hash}(\mathsf{a}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr})), \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
       $\mathtt{r}_{dr}))))$.
```

Fig. 37. The doctor process P'_{dr} (Fixing enforced prescribing-privacy with untappable channels with **s4'**).

```
let $P_{dr} =$
          $\mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})))$;
          $\mathsf{in}(\mathtt{ch}, (rcv\_Auth_{pt}, rcv\_PtProof))$;
          let $c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt})$ in
          let $(c\_Comt_{pt}, = c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)$ in
          if $\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt}) = \mathsf{true}$ then
          if $\mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt})) = \mathsf{true}$ then
          $\nu\mathtt{presc}$;
          $\nu\mathtt{r}_{dr}$;
          let $PrescriptID = \mathsf{hash}(\mathtt{presc}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))$ in
          $\mathsf{out}(\mathtt{ch}_{dp}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),$
                      $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),$
                      $(\mathtt{presc}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
                 $\mathtt{r}_{dr}))$.
```

Fig. 38. The doctor process P_{dr} (Fixing enforced prescribing-privacy with untappable channels with original assumptions).

```
$init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid main_{dr}\{\mathtt{A}/Id_{dr}, \mathsf{a}/presc\}^{\mathsf{chc}}) =$
$\mathtt{let}\ \mathtt{Id}_{dr} = \mathtt{A}\ \mathtt{in}$
$\nu\mathtt{Pnym}_{dr};$
$(!P_{dr} \mid$
$(\mathsf{out}(\mathtt{chc}, \mathtt{Id}_{dr});$
$\mathsf{out}(\mathtt{chc}, \mathtt{Pnym}_{dr}); \mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})));$
$\mathsf{in}(\mathtt{ch}, (rcv\_Auth_{pt}, rcv\_PtProof));$
$\mathsf{out}(\mathtt{chc}, (rcv\_Auth_{pt}, rcv\_PtProof));$
$\mathtt{let}\ c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt})\ \mathtt{in}$
$\mathtt{let}\ (c\_Comt_{pt}, = c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)\ \mathtt{in}$
$\mathtt{if}\ \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt}) = \mathtt{true}\ \mathtt{then}$
$\mathtt{if}\ \mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt})) = \mathtt{true}\ \mathtt{then}$
$\mathsf{out}(\mathtt{chc}, \mathsf{a});$
$\nu\mathtt{r}_{dr};$
$\mathsf{out}(\mathtt{chc}, \mathtt{r}_{dr});$
$\mathtt{let}\ PrescriptID = \mathsf{hash}(\mathsf{a}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))\ \mathtt{in}$
$\mathsf{out}(\mathtt{ch}_{dp}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),$
                $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),$
                $(\mathsf{a}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
        $\mathtt{r}_{dr}));$
$\mathsf{out}(\mathtt{chc}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),$
                $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),$
                $(\mathsf{a}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
        $\mathtt{r}_{dr})))).$
```

Fig. 39. The doctor process P_{dr}^{chc} (Fixing enforced prescribing-privacy with untappable channels with original assumptions).

```
$init_{dr}\{\mathtt{A}/Id_{dr}\}.(!P_{dr}\{\mathtt{A}/Id_{dr}\} \mid P'_{dr}\{\mathtt{A}/Id_{dr}\}) =$
$\mathtt{let}\ \mathtt{Id}_{dr} = \mathtt{A}\ \mathtt{in}$
$\nu\mathtt{Pnym}_{dr};$
$(!P_{dr} \mid$
$(\mathsf{out}(\mathtt{chc}, \mathtt{Id}_{dr}); \mathsf{out}(\mathtt{chc}, \mathtt{Pnym}_{dr}); \mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})));$
$\mathsf{in}(\mathtt{ch}, (rcv\_Auth_{pt}, rcv\_PtProof));$
$\mathsf{out}(\mathtt{chc}, (rcv\_Auth_{pt}, rcv\_PtProof));$
$\mathtt{let}\ c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt})\ \mathtt{in}$
$\mathtt{let}\ (c\_Comt_{pt}, = c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)\ \mathtt{in}$
$\mathtt{if}\ \mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt}) = \mathsf{true}\ \mathtt{then}$
$\mathtt{if}\ \mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt})) = \mathsf{true}\ \mathtt{then}$
$\mathsf{out}(\mathtt{chc}, \mathsf{a});$
$\nu\mathtt{r}_{dr};$
$\mathsf{out}(\mathtt{chc}, \mathtt{r}_{dr});$
$\mathtt{let}\ PrescriptID = \mathsf{hash}(\mathsf{b}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))\ \mathtt{in}$
$\mathsf{out}(\mathtt{ch}_{dp}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),$
                $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),$
                $(\mathsf{b}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
         $\mathtt{r}_{dr}));$
$\mathsf{out}(\mathtt{chc}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),$
                $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),$
                $(\mathsf{a}, \mathsf{hash}(\mathsf{a}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr})), \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
         $\mathtt{r}_{dr})))).$
```

Fig. 40. The doctor process P'_{dr} (Fixing enforced prescribing-privacy with untappable channels with original assumptions).

D Fixing enforced prescribing-privacy with chameleon bit-commitments

The following model is based on the model with assumption **s4’**

```
fun    ChCommit/2.
fun    fake/2.
reduc  chopen(ChCommit($m, r$), $r$) = $m$;
       chopen(ChCommit($m, r$), fake(ChCommit($m, r$), $n$)) = $n$.
```

Fig. 41. Additional functions and equations (Fixing enforced prescribing-privacy with chameleon bit-commitments).

The functions and equational theory consist of two parts as shown in Figure 41 and Figure 29. The framework of the DLVV08 protocol is the same as in Figure 30. Roles R_{dr}, R_{pt}, R_{ph}, R_{mpa} and R_{hii} are the same as in Figure 14, Figure 15, Figure 16, Figure 17 and Figure 18, respectively. Processes $P_{mpa}\_p_2$, and P_{hii} are the same as in Figure reffig:prompa2 and Figure reffig:prohii, respectively.

```
let $P_{dr}$ =
        $\nu$nonce$_{dr}$;
        out(ch, zk((Pnym$_{dr}$, Id$_{dr}$, nonce$_{dr}$), drcred(Pnym$_{dr}$, Id$_{dr}$, nonce$_{dr}$)));
        in(ch, ($rcv\_Auth_{pt}$, $rcv\_PtProof$));
        let $c\_Cred_{pt}$ = getpublic($rcv\_Auth_{pt}$) in
        let ($c\_Comt_{pt}$, = $c\_Cred_{pt}$) = getpublic($rcv\_PtProof$) in
        if Vfy-zk$_{\mathsf{Auth}_{pt}}$($rcv\_Auth_{pt}$, $c\_Cred_{pt}$) = true then
        if Vfy-zk$_{\mathsf{PtProof}}$($rcv\_PtProof$, ($c\_Comt_{pt}$, $c\_Cred_{pt}$)) = true then
        $\nu$presc;
        $\nu$r;
        let $commit_{pr}$ = ChCommit(presc, r) in
        $\nu$r$_{dr}$;
        let $PrescriptID$ = hash($commit_{pr}$, $c\_Comt_{pt}$, commit(Pnym$_{dr}$, r$_{dr}$)) in
        out(ch, (spk((Pnym$_{dr}$, r$_{dr}$, Id$_{dr}$, nonce$_{dr}$),
                    (commit(Pnym$_{dr}$, r$_{dr}$), drcred(Pnym$_{dr}$, Id$_{dr}$, nonce$_{dr}$)),
                    ($commit_{pr}$, $PrescriptID$, commit(Pnym$_{dr}$, r$_{dr}$), $c\_Comt_{pt}$)),
                r$_{dr}$));
        out(ch$_{dp}$, r).
```

Fig. 42. The doctor process P_{dr} (Fixing enforced prescribing-privacy with chameleon bit-commitments).

```
let $P_{pt}$ =
        in(ch, $rcv\_Auth_{dr}$);
        let $c\_Cred_{dr}$ = getpublic($rcv\_Auth_{dr}$) in
        if Vfy-zk$_{\mathsf{Auth}_{dr}}$($rcv\_Auth_{dr}$, $c\_Cred_{dr}$) = true then
        $\nu r_{pt}$;
        out(ch, (zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                  ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc)),
                zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                  (commit(Id$_{pt}$, $r_{pt}$),
                   ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc)))));
        in(ch, ($rcv\_PrescProof$, $rcv\_r_{dr}$));
        let ($rcv\_PrescCommit$, $c\_PrescriptID$, $c\_Comt_{dr}$, = commit(Id$_{pt}$, $r_{pt}$))
          = getSpkMsg($rcv\_PrescProof$) in
        if Vfy-spk$_{\mathsf{PrescProof}}$($rcv\_PrescProof$, ($c\_Cred_{dr}$, $rcv\_PrescCommit$, $c\_PrescriptID$,
                          $c\_Comt_{dr}$, commit(Id$_{pt}$, $r_{pt}$))) = true then
        let $c\_Pnym_{dr}$ = open($c\_Comt_{dr}$, $rcv\_r_{dr}$) in
        in(ch$_{dp}$, $xr$);
        let $c\_presc$ = chopen($rcv\_PrescCommit$, $xr$) in
        in(ch, $rcv\_Auth_{ph}$);
        if Vfy-sign($rcv\_Auth_{ph}$, $rcv_{pt}\_pk_{ph}$) = true then
        let (= $c_{pt}\_Id_{ph}$, $c_{pt}\_Id_{mpa}$) = getsignmsg($rcv\_Auth_{ph}$, $rcv_{pt}\_pk_{ph}$) in
        let $c_{pt}\_pk_{mpa}$ = key($c_{pt}\_Id_{mpa}$) in
        out(ch, zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc), Sss)));
        $\nu$nonce;
        let $vc_1$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Hii, $c_{pt}\_pk_{mpa}$))) in
        let $vc_2$ = zk(($c\_Pnym_{dr}$, $rcv\_r_{dr}$), ($rcv\_PrescProof$,
                     enc($c\_Pnym_{dr}$, $c_{pt}\_pk_{mpa}$))) in
        let $vc_3$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Pnym$_{pt}$, pk$_{sso}$))) in
        let $vc'_3$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Hii, pk$_{sso}$))) in
        let $vc_4$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Pnym$_{pt}$, $c_{pt}\_pk_{mpa}$))) in
        let $vc_5$ = zk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                    (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                     enc(Pnym$_{pt}$, $c_{pt}\_pk_{hii}$))) in
        let $c_5$ = enc($vc_5$, $c_{pt}\_pk_{mpa}$) in
        out(ch, ($rcv\_PrescProof$,
                spk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                   (ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc), commit(Id$_{pt}$, $r_{pt}$)),
                   nonce),
                $vc_1$, $vc_2$, $vc_3$, $vc'_3$, $vc_4$, $c_5$));
        out(ch$_{ptph}$, $xr$);
        in(ch, $rcv\_Invoice$);
        let $ReceiptAck$ = spk((Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                           ptcred(Id$_{pt}$, Pnym$_{pt}$, Hii, Sss, Acc),
                           ($c\_PrescriptID$, $c_{pt}\_Id_{ph}$, $vc_1$, $vc_2$, $vc_3$, $vc'_3$, $vc_4$, $c_5$)) in
        out(ch, $ReceiptAck$).
```

Fig. 43. The patient process P_{pt} (Fixing enforced prescribing-privacy with chameleon bit-commitments).

```
let $P_{ph} =$
        $\mathsf{out}(\mathsf{ch}, \mathsf{sign}((\mathtt{Id}_{ph}, c_{ph}\_Id_{mpa}), \mathtt{sk}_{ph}));$
        $\mathsf{in}(\mathsf{ch}, rcv\_PtAuthSss);$
        let $(c_{ph}\_Cred_{pt}, c_{ph}\_Sss) = \mathsf{getpublic}(rcv\_PtAuthSss)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{PtAuthSss}}(rcv\_PtAuthSss, (c_{ph}\_Cred_{pt}, c_{ph}\_Sss)) =$ true then
        $\mathsf{in}(\mathsf{ch}, (rcv_{ph}\_PrescProof, rcv_{ph}\_PtSpk,$
            $rcv\_vc_1, rcv\_vc_2, rcv\_vc_3, rcv\_vc'_3, rcv\_vc_4, rcv\_c_5));$
        let $(c_{ph}\_Comt_{dr}, c_{ph}\_Cred_{dr}) = \mathsf{getSpkVinfo}(rcv_{ph}\_PrescProof)$ in
        let $(rcv_{ph}\_PrescCommit, c_{ph}\_PrescriptID, = c_{ph}\_Comt_{dr}, c_{ph}\_Comt_{pt})$
          $= \mathsf{getSpkMsg}(rcv_{ph}\_PrescProof)$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(rcv_{ph}\_PrescProof, (c_{ph}\_Cred_{dr}, rcv_{ph}\_PrescCommit,$
                    $c_{ph}\_PrescriptID, c_{ph}\_Comt_{dr}, c_{ph}\_Comt_{pt})) =$ true then
        let $c\_msg = \mathsf{getSpkMsg}(rcv_{ph}\_PtSpk)$ in
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{PtSpk}}(rcv_{ph}\_PtSpk,$
                    $(c_{ph}\_Cred_{pt}, c_{ph}\_Comt_{pt}, c\_msg)) =$ true then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_1) = \mathsf{getpublic}(rcv\_vc_1)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv\_vc_1, (c_{ph}\_Cred_{pt}, c\_Enc_1, rcv_{ph}\_pk_{mpa})) =$ true then
        let $(= rcv_{ph}\_PrescProof, c\_Enc_2) = \mathsf{getpublic}(rcv\_vc_2)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(rcv\_vc_2, (rcv_{ph}\_PrescProof,$
                            $c\_Enc_2, rcv_{ph}\_pk_{mpa})) =$ true then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_3) = \mathsf{getpublic}(rcv\_vc_3)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv\_vc_3, (c_{ph}\_Cred_{pt}, c\_Enc_3, \mathsf{pk}_{sso})) =$ true then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc'_3) = \mathsf{getpublic}(rcv\_vc'_3)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv\_vc'_3, (c_{ph}\_Cred_{pt}, c\_Enc'_3, \mathsf{pk}_{sso})) =$ true then
        let $(= c_{ph}\_Cred_{pt}, c\_Enc_4) = \mathsf{getpublic}(rcv\_vc_4)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv\_vc_4,$
                      $(c_{ph}\_Cred_{pt}, c\_Enc_4, rcv_{ph}\_pk_{mpa})) =$ true then
        $\mathsf{in}(\mathsf{ch}_{ptph}, rcv_{ph}\_xr);$
        let $c_{ph}\_presc = \mathsf{chopen}(rcv_{ph}\_PrescCommit, rcv_{ph}\_xr)$ in
        $\mathsf{out}(\mathsf{ch}, \mathsf{invoice}(c_{ph}\_PrescriptID));$
        $\mathsf{in}(\mathsf{ch}, rcv\_ReceiptAck);$
        if $\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv\_ReceiptAck, (c_{ph}\_Cred_{pt}, c_{ph}\_PrescriptID,$
        $\mathtt{Id}_{ph}, rcv\_vc_1, rcv\_vc_2, rcv\_vc_3, rcv\_vc'_3, rcv\_vc_4, rcv\_c_5)) =$ true then
        $\mathsf{out}(\mathsf{ch}, (\mathsf{sign}((\mathtt{Id}_{ph}, c_{ph}\_Id_{mpa}), \mathtt{sk}_{ph}), \mathtt{Id}_{ph}));$
        $\mathsf{in}(\mathsf{ch}, rcv\_Auth_{mpa});$
        if $\mathsf{Vfy\text{-}sign}(rcv\_Auth_{mpa}, rcv_{ph}\_pk_{mpa}) =$ true then
        $\mathsf{out}(\mathsf{ch}, (rcv_{ph}\_PrescProof,$
              $rcv\_vc_1, rcv\_vc_2, rcv\_vc_3, rcv\_vc'_3, rcv\_vc_4, rcv\_c_5,$
              $rcv\_ReceiptAck))$
        $\mathsf{out}(\mathsf{ch}_{pm}, rcv_{ph}\_xr).$
```

Fig. 44. The pharmacist process P_{ph} (Fixing enforced prescribing-privacy with chameleon bit-commitments).

```
let $P_{mpa}\_p_1 =$
                in($\mathsf{ch}, (rcv_{mpa}\_Auth_{ph}, c_{mpa}\_Id_{ph})$);
                let $rcv_{mpa}\_pk_{ph} = \mathsf{key}(c_{mpa}\_Id_{ph})$ in
                $\mathsf{Vfy\text{-}sign}(rcv_{mpa}\_Auth_{ph}, rcv_{mpa}\_pk_{ph}) = \mathsf{true}$
                let $(= c_{mpa}\_Id_{ph}, = \mathsf{Id}_{mpa})$
                   $= \mathsf{getSpkMsg}(rcv_{mpa}\_Auth_{ph}, rcv_{mpa}\_pk_{ph})$ in
                $\mathsf{out}(\mathsf{ch}, \mathsf{sign}(\mathsf{Id}_{mpa}, \mathsf{sk}_{mpa}))$;
                $\mathsf{in}(\mathsf{ch}, (rcv_{mpa}\_PrescProof, rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2, rcv_{mpa}\_vc_3,$
                        $rcv_{mpa}\_vc'_3, rcv_{mpa}\_vc_4, rcv_{mpa}\_c_5, rcv_{mpa}\_ReceiptAck))$;
                let $(c_{mpa}\_Comt_{dr}, c_{mpa}\_Cred_{dr}) = \mathsf{getSpkVinfo}(rcv_{mpa}\_PrescProof)$ in
                let $(rcv_{mpa}\_PrescCommit, c_{mpa}\_PrescriptID, = c_{mpa}\_Comt_{dr}, c_{mpa}\_Comt_{pt})$
                   $= \mathsf{getSpkMsg}(rcv_{mpa}\_PrescProof)$ in
                if $\mathsf{Vfy\text{-}spk}_{\mathsf{PrescProof}}(rcv_{mpa}\_PrescProof, (c_{mpa}\_Cred_{dr}, rcv_{mpa}\_PrescCommit,$
                                 $c_{mpa}\_PrescriptID, c_{mpa}\_Comt_{dr}, c_{mpa}\_Comt_{pt})) = \mathsf{true}$ then
                let $(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_1) = \mathsf{getpublic}(rcv_{mpa}\_vc_1)$ in
                if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv_{mpa}\_vc_1,$
                                 $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_1, \mathsf{pk}_{mpa})) = \mathsf{true}$ then
                let $c_{mpa}\_Hii = \mathsf{dec}(c_{mpa}\_Enc_1, \mathsf{sk}_{mpa})$ in
                let $(= rcv_{mpa}\_PrescProof, c_{mpa}\_Enc_2) = \mathsf{getpublic}(rcv_{mpa}\_vc_2)$ in
                if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncDrnymMpa}}(rcv_{mpa}\_vc_2,$
                                      $(rcv_{mpa}\_PrescProof, c_{mpa}\_Enc_2, \mathsf{pk}_{mpa})) = \mathsf{true}$ then
                let $c_{mpa}\_Pnym_{dr} = \mathsf{dec}(c_{mpa}\_Enc_2, \mathsf{sk}_{mpa})$ in
                let $(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc3) = \mathsf{getpublic}(rcv_{mpa}\_vc_3)$ in
                if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv_{mpa}\_vc_3,$
                                    $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_3, \mathsf{pk}_{sso})) = \mathsf{true}$ then
                let $(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc'_3) = \mathsf{getpublic}(rcv_{mpa}\_vc'_3)$ in
                if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncHii}}(rcv_{mpa}\_vc'_3,$
                                 $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc'_3, \mathsf{pk}_{sso})) = \mathsf{true}$ then
                let $(= c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_4) = \mathsf{getpublic}(rcv_{mpa}\_vc_4)$ in
                if $\mathsf{Vfy\text{-}zk}_{\mathsf{VEncPtnym}}(rcv_{mpa}\_vc_4,$
                                    $(c_{mpa}\_Cred_{pt}, c_{mpa}\_Enc_4, \mathsf{pk}_{mpa})) = \mathsf{true}$ then
                let $c_{mpa}\_Pnym_{pt} = \mathsf{dec}(c_{mpa}\_Enc_4, \mathsf{sk}_{mpa})$ in
                if $\mathsf{Vfy\text{-}spk}_{\mathsf{ReceiptAck}}(rcv_{mpa}\_ReceiptAck, (c_{mpa}\_Cred_{pt},$
                     $c_{mpa}\_PrescriptID, c_{mpa}\_Id_{ph}, rcv_{mpa}\_vc_1, rcv_{mpa}\_vc_2, rcv_{mpa}\_vc_3,$
                     $rcv_{mpa}\_vc'_3, rcv_{mpa}\_vc_4, rcv_{mpa}\_c_5))$
                  $= \mathsf{true}$ then
                $\mathsf{in}(\mathsf{ch}_{pm}, rcv_{mpa}\_xr)$;
                let $c_{mpa}\_presc = \mathsf{chopen}(rcv_{mpa}\_PrescCommit, rcv_{mpa}\_xr)$ in0.
```

Fig. 45. The MPA process P_{mpa} in Pharmacist-MPA sub-protocol (Fixing enforced prescribing-privacy with chameleon bit-commitments).

```
let $P'_{dr}$ =
        $\mathsf{out}(\mathtt{ch}, \mathsf{zk}((\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})));$
        $\mathsf{in}(\mathtt{ch}, (rcv\_Auth_{pt}, rcv\_PtProof));$
        $\mathsf{in}(\mathtt{chc}, (rcv\_Auth_{pt}, rcv\_PtProof));$
        let $c\_Cred_{pt} = \mathsf{getpublic}(rcv\_Auth_{pt})$ in
        let $(c\_Comt_{pt}, = c\_Cred_{pt}) = \mathsf{getpublic}(rcv\_PtProof)$ in
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{Auth}_{pt}}(rcv\_Auth_{pt}, c\_Cred_{pt}) = \mathtt{true}$ then
        if $\mathsf{Vfy\text{-}zk}_{\mathsf{PtProof}}(rcv\_PtProof, (c\_Comt_{pt}, c\_Cred_{pt})) = \mathtt{true}$ then
        $\nu \mathtt{presc}';$
        $\mathsf{out}(\mathtt{chc}, \mathtt{presc});$
        $\nu \mathtt{r};$
        let $commit_{pr} = \mathsf{ChCommit}(\mathtt{presc}, \mathtt{r})$ in
        $\mathsf{out}(\mathtt{chc}, \mathsf{fake}(commit_{pr}, \mathtt{presc}));$
        $\nu \mathtt{r}_{dr};$
        $\mathsf{out}(\mathtt{chc}, \mathtt{r}_{dr});$
        let $PrescriptID = \mathsf{hash}(commit_{pr}, c\_Comt_{pt}, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}))$ in
        $\mathsf{out}(\mathtt{ch}, (\mathsf{spk}((\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}, \mathtt{Id}_{dr}),$
                    $(\mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), \mathsf{drcred}(\mathtt{Pnym}_{dr}, \mathtt{Id}_{dr})),$
                    $(commit_{pr}, PrescriptID, \mathsf{commit}(\mathtt{Pnym}_{dr}, \mathtt{r}_{dr}), c\_Comt_{pt})),$
              $\mathtt{r}_{dr}));$
        $\mathsf{out}(\mathtt{ch}_{dp}, \mathtt{r}).$
```

Fig. 46. The doctor process P'_{dr} (Fixing enforced prescribing-privacy with chameleon bit-commitments).